ELSEVIER
Theoretical Computer Science 282 (2002) 101–150
Theoretical Computer Science
www.elsevier.com/locate/tcs

Automatic verification of real-time systems with discrete probability distributions ☆

Marta Kwiatkowska[a, *], Gethin Norman[a], Roberto Segala[b], Jeremy Sproston[a]

[a] *School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK*
[b] *Dipartimento di Informatica, Università di Verona, Verona, Italy*

Abstract

We consider the timed automata model of Alur and Dill (Theoret. Comput. Sci. 126 (1994) 183–235), which allows the analysis of real-time systems expressed in terms of quantitative timing constraints. Traditional approaches to real-time system description express the model purely in terms of nondeterminism; however, it is often desirable to express the likelihood of the system making certain transitions. In this paper, we present a model for real-time systems augmented with discrete probability distributions. Furthermore, two approaches to model checking are introduced for this model. The first uses the algorithm of Baier and Kwiatkowska (Distributed Comput. 11 (1998) 125–155) to provide a verification technique against temporal logic formulae which can refer both to timing properties and probabilities. The second, generally more efficient, technique concerns the verification of probabilistic, real-time reachability properties.

1. Introduction

The proliferation of digital technology embedded into real-life environments has led to increased interest in computer systems expressed in terms of quantitative timing constraints. Examples of such *real-time systems* include communication protocols, digital circuits with uncertain delay lengths, and media synchronization protocols. A number of frameworks exist for modelling, analysis and verification of such systems. A formalism that has received much attention, both in terms of theoretical and practical developments, is that of *timed automata*; in particular, the theory of automatically verifying timed automata against properties expressed in a real-time temporal logic is advanced, and is supported by a number of tools [7, 10].

Traditional approaches to the formal description of real-time systems express the system model purely in terms of non-determinism. However, it may be desirable to

☆ Supported in part by EPSRC grants GR/M04617, GR/M13046, and GR/N22960.
* Corresponding author. Tel.: +44-121-414-4773; fax: +44-121-414-4281.
E-mail address: m.z.kwiatkowska@cs.bham.ac.uk (M. Kwiatkowska).

0304-3975/02/$ - see front matter
PII: S0304-3975(01)00046-9

express the relative likelihood of the system exhibiting certain behaviour. For example, we may wish to model a system for which the likelihood of a certain event occurring changes with respect to the amount of time elapsed. This notion is particularly important when considering fault-tolerant systems. Furthermore, we may also wish to refer to the likelihood of certain properties being satisfied by the real-time system, and to have a model checking algorithm for verifying the truth of these assertions. The remit of this paper is to address these problems.

We present a model for real-time systems that are described partially in terms of discrete probability distributions, and two automatic verification methods for this model against probabilistic timed properties. The system model is called a *probabilistic timed automaton*, and differs from the traditional timed automaton model of [2, 3] in the following respect: the edge relation of probabilistic timed automaton is both nondeterministic and probabilistic in nature. More precisely, instead of making a purely nondeterministic choice over the set of currently enabled edges, we choose amongst the set of enabled discrete probability distributions, each of which is defined over a finite set of edges. We then make a probabilistic choice as to which edge to take according to the selected distribution. As with usual timed automata techniques, the underlying model of time is assumed to be *dense*; that is, the time domain is modelled by the reals ($\mathbb{R}$) or rationals ($\mathbb{Q}$).

In addition, we require appropriate languages to specify properties of probabilistic real-time systems. Firstly, a specification language commonly used for stating real-time system requirements, timed computation tree logic (TCTL) [21], is adapted to cater for probability. A common approach taken in probabilistic temporal logics is to add the probabilistic operator $[\cdot]_{\geqslant p}$ where p is a probability bound. For example, $[\phi_1 \; \exists\mathscr{U} \; \phi_2]_{\geqslant p}$ is true if the probability of $\phi_1 \; \exists\mathscr{U} \; \phi_2$ is at least p. Therefore, we develop our specification language, probabilistic timed computation tree logic (PTCTL), by adding such probabilistic operators to TCTL. The resulting logic allows us to express such quality of service properties as "with probability 0.7, there will be a response between 5 and 7 time units after a query". Secondly, we also consider the less expressive class of *reachability* properties, an example of which is "with probability 0.99 or greater, a data packet is delivered within 5 time units".

The denseness of the time domain means that the state space of timed automata is infinite. Therefore, automatic verification of timed automata is performed by constructing a finite-state quotient of the system model. This quotient takes the form of a state-labelled transition system which represents all of the timed automaton's behaviours, and which can be analysed using analogues of traditional model checking techniques. We adopt this method in order to construct a finite quotient of probabilistic timed automata, and note that certain desirable properties required for model checking in the nonprobabilistic case are preserved in our context. The addition of discrete probability distributions to timed automata means that the transitions of the resulting finite quotient structure are both nondeterministic and probabilistic in nature, and therefore the model checking methods employed must accommodate this characteristic. For the case in which PTCTL properties are considered, the verification algorithms of [6] are

used for this purpose. However, they are defined with respect to probabilistic branching time logic (PBTL), which does not allow the expression of dense timing constraints. Hence, we present a method for translating a given PTCTL formula into a corresponding PBTL formula. The model checking algorithm of [6] is then used to verify the PBTL properties over our probabilistic–nondeterministic quotient structure, the results of which allow us to conclude whether the original probabilistic timed automaton satisfies its PTCTL specification. In the case of such temporal logic properties, the quotient structure is obtained by application of the *region equivalence* of [3].

Unfortunately, for many real-life examples of timed systems, region equivalence induces an unnecessarily fine granularity on the infinite state space of system model, making model checking impractical. Therefore, we also consider probabilistic real-time reachability properties, which, while lacking the expressive power of PTCTL, can express a number of useful requirements. Consideration of such a narrower class of properties means that we can abstract more information when constructing the quotient structure of the probabilistic timed automaton, and therefore the size of the quotient can be smaller. Our approach is to adapt the *forward reachability* algorithms of [13, 24, 27] for timed automata to cater for probability. Unfortunately, the adaptation necessitates the loss of an appealing characteristic of these algorithms, namely that they are *on-the-fly* (that is, the algorithms may terminate as soon as an answer to the verification problem is found, without searching exhaustively through the state space). Furthermore, the probability of reachability that we can compute is not exact, but is instead an *upper* bound on the true reachability probability. However, our approach results in a quotient structure which is no larger than that obtained via the associated region construction, and which may, in practice, be significantly smaller. The probability bound obtained by our method is useful in certain contexts; for example, in the case of invariance properties for which we are interested in the probability of reaching an 'unsafe' state.

An example of a real-time system which could be subject to these techniques is the bounded retransmission protocol, which is modelled as a network of purely nondeterministic timed automata in [12]. Each communication channel is represented as a timed automaton which features a nondeterministic choice over two edges, one of which corresponds to the correct transmission of the message, the other to the message's loss. Using our framework, the relative likelihood of such a loss occurring could be represented by replacing this nondeterministic choice by a probabilistic choice between the two edges; for example, a probabilistic timed automaton could be used to model that a message is lost with probability 0.05 each time a communication channel is used. Similarly, the system requirements of the bounded retransmission protocol could be expanded to admit reasoning about the probability of certain system behaviours. For instance, we may require that, with probability at least 0.99, any data chunk transmitted by the sender is successfully processed by the receiver within 10 time units.

The model presented in this paper has similarities with other frameworks for probabilistic real-time systems. In particular, the approach of [18] is also to augment timed automata with discrete probability distributions; however, these distributions are obtained by normalization of edge-labelling weights. Furthermore, the model checking

algorithm of [18] is with respect to an action-based logic, rather than a state-based logic such as PTCTL. A dense time, automata-based model with discrete and *continuous* probability distributions is presented in [1], along with a quotient construction and TCTL model checking method similar to that of [2]. However, the model of [1] does not permit any nondeterministic choice, and its use of continuous probability distributions, while a highly expressive modelling mechanism, does not permit the model to be automatically verified against logics which include bounds on probability. Furthermore, note that the temporal logic of [19] has syntactic similarities with the logic PTCTL, although this former logic is interpreted with respect to discrete, not dense time.

The paper proceeds as follows. Section 2 introduces some preliminary concepts and notation relating to probability distributions, execution sequences and dense state spaces. Probabilistic timed automata are defined in Section 3 as our model for probabilistic–nondeterministic real-time systems, and an example of how the framework can be used to model a simple communication protocol is presented. Section 4 presents the underlying model of probabilistic timed automata, which are used to interpret formulae of the logic, PTCTL, introduced in Section 5. Section 6 explores the PTCTL model checking problem for probabilistic timed automata, and presents a finite-state quotient construction for this model, a method for translating a PTCTL formula into an equivalent PBTL formula, and finally a verification method. Section 7 considers the verification of probabilistic timed automata against reachability properties. To conclude, Section 8 analyses the complexity of the model checking techniques, and suggests further directions of research. A preliminary version of this paper appeared as [22].

2. Preliminaries

Probability distributions: We denote the set of (finite) discrete probability distributions over a set S by $\mu(S)$. Therefore, each $p \in \mu(S)$ is a function $p : S \to [0,1]$ such that $\sum_{s \in S} p(s) = 1$ and the set $\{s \mid s \in S \text{ and } p(s) > 0\}$ is finite.

Markov decision processes: A Markov decision process is a tuple $(Q, Steps)$, where Q is a set of states, and $Steps : Q \to 2^{\mu(Q)}$ is a function assigning a set of probability distributions to each state. Our intuition is that the Markov decision process traverses the state space by making transitions determined by *Steps*; that is, in the state q, a transition is made by first nondeterministically selecting a probability distribution p from the set $Steps(q)$, and then performing a probabilistic choice according to p as to which state to move to. If the state selected by p is q', then we denote such a transition by $q \xrightarrow{p} q'$. Throughout the paper, we present variants of Markov decision processes as necessary. For example, occasionally we will require an additional "event" in the definition of *Steps*, so that, for some set Σ, $Steps : Q \to 2^{\Sigma \times \mu(Q)}$ is now a function assigning a pair (σ, p), comprising of an event and a probability distribution, to each state (a transition is now denoted by $q \xrightarrow{\sigma,p} q'$). Furthermore, we often require state labelling functions of the form $L : Q \to 2^{AP}$ to be included in the definition of a Markov decision process. Such functions assign a set of atomic propositions from the set AP

to each state. Then, in this case, a (labelled) Markov decision process is a tuple $(Q, Steps, L)$.

Paths: Labelled paths (or execution sequences) are nonempty finite or infinite sequences of the form

$$\omega = q_0 \xrightarrow{l_0} q_1 \xrightarrow{l_1} q_2 \xrightarrow{l_2} \cdots,$$

where q_i are states and l_i are labels for transitions. We use the following notation for such paths. Take any path ω. Then the first state of ω is denoted by $first(\omega)$. If ω is finite then the last state of ω is denoted by $last(\omega)$. The length of a path, $|\omega|$, is defined in the usual way: if ω is the finite path $\omega = q_0 \xrightarrow{l_0} q_1 \xrightarrow{l_1} \cdots \xrightarrow{l_{n-1}} q_n$, then $|\omega| = n$; if ω is an infinite path, then we let $|\omega| = \infty$. If $k \leqslant |\omega|$ then $\omega(k)$ denotes the kth state of ω and $step(\omega, k)$ is the label of the kth step (that is, $\omega(k) = q_k$ and $step(\omega, k) = l_k$). $\omega^{(k)}$ is the kth prefix of ω; that is, if $k < |\omega|$ then $\omega^{(k)} = q_0 \xrightarrow{l_0} q_1 \xrightarrow{l_1} \cdots \xrightarrow{l_{k-1}} q_k$, and if $k \geqslant |\omega|$ then $\omega^{(k)} = \omega$. If $\omega = q_0 \xrightarrow{l_0} q_1 \xrightarrow{l_1} \cdots \xrightarrow{l_{n-1}} q_n$ is a finite path and $\omega' = q'_0 \xrightarrow{l'_0} q'_1 \xrightarrow{l'_1} \cdots$ is a finite or infinite path with $last(\omega) = first(\omega')$, then we let the *concatenation* of ω and ω' be

$$\omega\omega' = q_0 \xrightarrow{l_0} q_1 \xrightarrow{l_1} q_2 \cdots \xrightarrow{l_{n-1}} q_n \xrightarrow{l'_0} q'_1 \xrightarrow{l'_1} \cdots .$$

Clocks and clock valuations: A *clock* is a real-valued variable which increases at the same rate as real time. Let $\mathscr{X} = \{x_1, \ldots, x_n\}$ be a set of clocks, and let $v : \mathscr{X} \to \mathbb{R}$ be a function assigning a real value to each of the clocks in this set. Such a function is called a *clock valuation*. We denote the set of all clock valuations of $\mathscr{X}$ by $\mathbb{R}^{\mathscr{X}}$. Let $\mathbf{0}$ be the clock valuation that assigns 0 to all clocks in $\mathscr{X}$. For some $X \subseteq \mathscr{X}$, we write $v[X := 0]$ for the clock valuation that assigns 0 to clocks in X, and agrees with v for all clocks in $\mathscr{X} \setminus X$ (that is, $\forall x \in X. v[X := 0](x) = 0$ and $\forall x \in \mathscr{X} \setminus X. v[X := 0](x) = v(x)$). Informally, we write $v[x := 0]$ if X contains the single clock x. In addition, for every $t \in \mathbb{R}$, $v + t$ denotes the clock valuation for which all clocks $x \in \mathscr{X}$ take the value $v(x) + t$.

Constraints: A *constraint* over $\mathscr{X}$ is an expression of the form $x_i \sim c$ or $x_i - x_j \sim c$, where $1 \leqslant i \neq j \leqslant n$, $\sim \in \{<, \leqslant, \geqslant, >\}$ and $c \in \mathbb{N} \cup \{\infty\}$. By convention (see [14]), a "dummy" variable x_0, which is always taken to represent 0, is assumed. The presence of x_0 allows all constraints to be written uniformly as $x_i - x_j \sim c$ where $0 \leqslant i \neq j \leqslant n$, $\sim \in \{<, \leqslant\}$ and $c \in \mathbb{Z} \cup \{\infty\}$.

A clock valuation v *satisfies* a constraint $x_i - x_j \sim c$ iff $v(x_i) - v(x_j) \sim c$.

Zones: A *zone* of $\mathscr{X}$, written ζ, is a convex subset of the valuation space $\mathbb{R}^{\mathscr{X}}$ described by a conjunction of constraints. Formally, a zone ζ is the set of valuations which satisfy the conjunction of $n \cdot (n+1)$ constraints given by

$$\bigwedge_{0 \leqslant i \neq j \leqslant n} x_i - x_j \sim_{i,j} c_{i,j}.$$

Let $\mathbf{Z}_{\mathscr{X}}$ be the set of all zones of $\mathscr{X}$. We denote by $c_{\max}(\zeta)$ the largest constant used in the description of a zone; that is, $c_{\max}(\zeta) = \max\{|c_{i,j}| \mid 0 \leqslant i \neq j \leqslant n\}$, where $|c|$ is the absolute value of c if $c \in \mathbb{Z}$ and 0 if $c = \infty$.

Observe that more than one conjunction of constraints may correspond to the same subset of $\mathbb{R}^{\mathscr{X}}$. Therefore, in the sequel, we only consider zones that are in *canonical form*; that is, when their constraints are as 'tight' as possible (see [17] for an $O(n^3)$ algorithm to achieve this for any zone). Such a canonical form allows us to use the *semantic* interpretation of zones as sets of valuations interchangeably with the original, *syntactic* interpretation of zones as conjunctions of constraints. Observe that this means that semantic equality can be reduced to syntactic equality. Other semantic operations, such as intersection $\zeta_1 \cap \zeta_2$, are well-defined operations on polyhedra such as zones, and have their corresponding syntactic transformations. However, we generally use the semantic interpretation throughout this paper.

In the following definition we introduce the clock reset operation on zones.

Definition 1. For any zones $\zeta, \zeta' \in \mathbf{Z}_{\mathscr{X}}$, and a set of clocks $X \subseteq \mathscr{X}$, let $\zeta[X := 0] = \{v[X := 0] \mid v \in \zeta\}$.

We write $\zeta_{\mathbf{0}}$ for the zone which contains the single clock valuation $\mathbf{0}$. Let $\zeta \in \mathbf{Z}_{\mathscr{X}}$ be a zone and $v \in \mathbb{R}^{\mathscr{X}}$ be a valuation. Given that we return to our syntactic view of zones as conjunctions of constraints, $\zeta[v]$ is the boolean value obtained by replacing each occurrence of a clock $x \in \mathscr{X}$ in ζ by $v(x)$. If $\zeta[v] = \texttt{true}$ then we say that v *satisfies* ζ, also denoted by $v \in \zeta$ (such set membership notation tallies with the semantic view of zones as sets of valuations).

3. Probabilistic timed automata

This section introduces *probabilistic timed automata* as a modelling framework for real-time systems with probability; this formalism is derived from timed automata [2, 3]. Here, we extend timed automata with discrete probability distributions over edges, so that the choice of the next location of the automaton is now probabilistic, in addition to nondeterministic, in nature. Furthermore, we incorporate *invariant conditions* [21] into the probabilistic timed automaton in order to enforce upper bounds on the time at which certain probabilistic choices are made.

Definition 2 (*Probabilistic timed automaton*). A probabilistic timed automaton is a tuple $G = (\mathscr{S}, \mathscr{L}, \bar{s}, \mathscr{X}, inv, prob, \langle \tau_s \rangle_{s \in \mathscr{S}})$ which contains:

- a finite set $\mathscr{S}$ of nodes,
- a function $\mathscr{L} : \mathscr{S} \to 2^{\mathrm{AP}}$ assigning to each node of the automaton the set of atomic propositions that are true in that node,
- a start node $\bar{s} \in \mathscr{S}$,
- a finite set $\mathscr{X}$ of clocks,

- a function $inv : \mathscr{S} \to \mathbf{Z}_{\mathscr{X}}$ assigning to each node an invariant condition,
- a function $prob : \mathscr{S} \to \mathscr{P}_{\mathrm{fn}}(\mu(\mathscr{S} \times 2^{\mathscr{X}}))$ assigning to each node a (finite, nonempty) set of discrete probability distributions on $\mathscr{S} \times 2^{\mathscr{X}}$,
- a family of functions $\langle \tau_s \rangle_{s \in \mathscr{S}}$ where, for any $s \in \mathscr{S}$, $\tau_s : prob(s) \to \mathbf{Z}_{\mathscr{X}}$ assigns to each $p \in prob(s)$ an enabling condition.

The behaviour of the system takes the form of transitions between states, which can be as a result of either the passage of time or the execution of a discrete transition. The role of the invariant condition is to describe the set of *admissible states* of the probabilistic timed automaton; therefore, we forbid transitions to inadmissible states.

The system starts in node $\bar{s}$ with all of its clocks initialized to 0. The values of all the clocks increase uniformly with time. At any point in time, if the system is in node s and the invariant condition will no longer be satisfied by letting any time advance, then the system can either (a) remain in its current node and let time advance, or (b) make a *discrete transition* if there exists a distribution $p \in prob(s)$ whose corresponding enabling condition $\tau_s(p)$ is satisfied by the current values of the clocks. Alternatively, if the invariant condition will be violated by letting time advance then the system must make a discrete transition. Discrete transitions are instantaneous and consist of the following two steps performed in succession: firstly, the system makes a *nondeterministic choice* between the set of distributions $p \in prob(s)$ whose corresponding enabling condition $\tau_s(p)$ is satisfied by the current values of the clocks. Secondly, supposing that the probability distribution p is chosen, the system then makes a *probabilistic transition* according to p; that is, for any $s' \in \mathscr{S}$ and $X \subseteq \mathscr{X}$, the probability that the system will make a state transition to node s', and reset all the clocks in X to 0, is given by $p(s', X)$.

For simplicity, the invariant and enabling conditions are subject to the following assumptions: first, if, in some state in the execution of G, allowing any amount of time to elapse would violate the invariant condition of the current node, then the enabling condition of at least one probability distribution is satisfied.[1] Second, we assume that it is never possible to perform a discrete transition to a node for which the invariant condition is not satisfied by the current values of the clocks. Formally, for each node $s \in \mathscr{S}$ and each distribution of that node $p \in prob(s)$, assume that, for all $s' \in \mathscr{S}$, $X \subseteq \mathscr{X}$ such that $p(s', X) > 0$, it is the case that $\tau_s(p)[X := 0] \subseteq inv(s')$. We refer to this assumption as that of *admissible targets*.

We use $\mathbf{Z}_{\mathscr{X}}(G)$ to denote the set of all zones appearing as an invariant or enabling condition of G. Formally, let

$$\mathbf{Z}_{\mathscr{X}}(G) = \{inv(s) \in \mathbf{Z}_{\mathscr{X}} \mid s \in \mathscr{S}\} \cup \{\tau_s(p) \in \mathbf{Z}_{\mathscr{X}} \mid s \in \mathscr{S} \text{ and } p \in prob(s)\}.$$

[1] Another solution is to identify an additional discrete probability distribution $p_s^{inv} \in \mu(\mathscr{S} \times 2^{\mathscr{X}})$ with each $s \in \mathscr{S}$, which becomes enabled in s at the points for which progression of any amount of time would violate the node's invariant $inv(s)$.

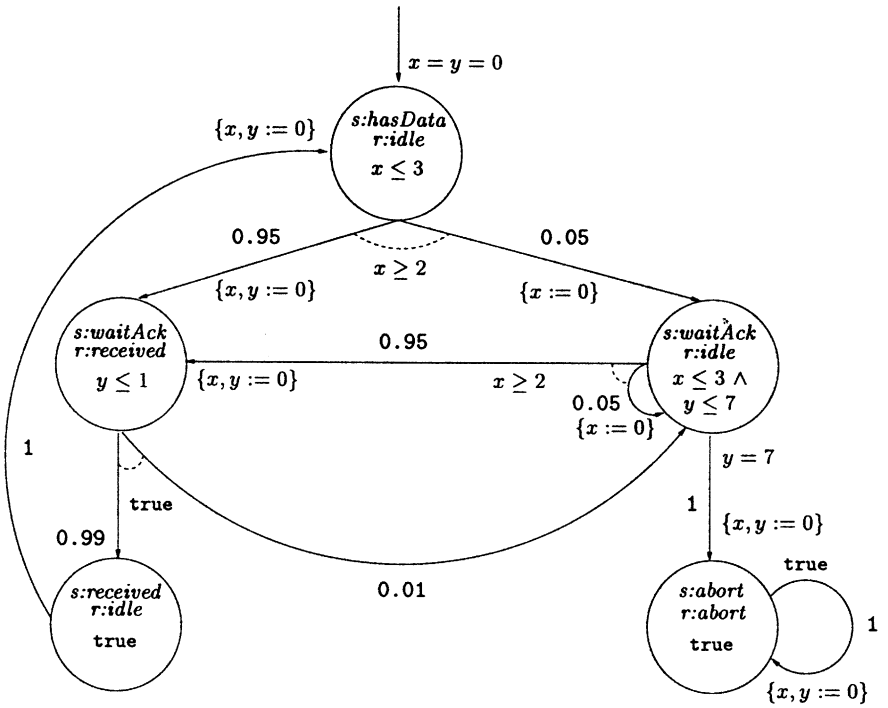

Fig. 1. The probabilistic timed automaton G_1.

Furthermore, let $c_{\max}(G) = \max\{c_{\max}(\zeta) \mid \zeta \in \mathbf{Z}_{\mathscr{X}}(G)\}$ be the maximal constant used in the description of G. In later sections, this constant will be used to ensure the decidability of our model checking methods.

3.1. Example

An example of a probabilistic timed automaton which models a simple communication protocol operating with an unreliable, lossy channel is shown in Fig. 1. The system consists of a sender and a receiver which communicate over a single channel, and two global clocks x and y (we make the assumption of the existence of global clocks for simplicity). Transit across the medium is instantaneous. Operation of the protocol commences with the delivery of new data to the sender, and with both clocks x and y set to 0. The sender retains the data for between 2 and 3 time units before transmitting it onto the medium and resetting the clock x to 0. With probability 0.95, the data is correctly received, in which case the receiver waits for not more than 1 time unit to attempt to return an acknowledgement to the sender. This attempt is successful with probability 0.99, after which an arbitrary length of time may elapse before another data packet is delivered to the sender for transmission across the medium. However, if either the acknowledgement or the original data packet was

lost by the channel, then the receiver is idle, whereas the sender is still waiting for an acknowledgement; therefore, the sender attempts to re-send the data at a frequency varying between 2 and 3 time units. If exactly 7 time units have elapsed since the data first arrived at the sender, or the receiver last received the data, then the system aborts.

This communication protocol is modelled by the probabilistic timed automaton G_1 in the following way. Consider the initial node, $s:hasData$, $r:idle$, henceforth abbreviated to hi. The invariant of hi is the zone described by the constraint written in the body of the node, $x \leqslant 3$, which represents the fact that the clock x is not allowed to exceed 3. The node hi has one probability distribution associated with it, as represented by the edges connected by a dashed arc, which is only enabled when $x \geqslant 2$. Therefore, when x is between 2 and 3, this distribution, which corresponds to the sender transmitting the data onto the medium, may be nondeterministically chosen. Furthermore, if x equals 3 then the invariant condition requires that the distribution *must* be taken. This distribution is defined over two edges: the first leads to $s:waitAck$, $r:received$ (abbreviated to wr), and corresponds to the successful delivery of the data across the medium; as such, it is labelled with the probability 0.95. The second edge leads to $s:waitAck$, $r:idle$ (abbreviated to wi), and represents loss of data with probability 0.05. If the former edge is taken, both clocks x and y are reset, whereas, in the latter case, only x is reset, as denoted by the edge labels $\{x,y:=0\}$ and $\{x:=0\}$, respectively. More formally, $p \in prob(hi)$, where $p(wr,\{x,y\})=0.95$, $p(wi,\{x\})=0.05$. Another aspect of probabilistic timed automaton behaviour can be witnessed in the node wi: it is possible for *both* of the distributions associated with wi, $p'_1, p'_2 \in prob(wi)$, to be enabled at the same time. In such a case, there can be a nondeterministic choice between choosing p'_1 and choosing p'_2. From this description of the behaviour in the nodes hi and wi, the behaviour of G_1 in other nodes follows in a straightforward manner. The reader can verify that G_1 does indeed model the communication protocol described in the previous paragraph. In subsequent sections, we will refer back to this example, using the abbreviations ri for $s:received$, $r:idle$, and aa for $s:abort$, $r:abort$.

In addition to considering the way in which the communication protocol can be modelled, we must also identify a set of requirements which we wish the protocol to satisfy, and to formalize them in an appropriate manner. Their validity with respect to the probabilistic timed automaton G_1 could then be verified using the model checking techniques presented in the remainder of this paper (although, as we will see later, not all of these properties can be verified by both of our presented approaches). Four types of requirements are now presented, along with examples relevant to the communication protocol.

Reachability: The system can reach a certain set of states with a given probability. For example, "with probability 0.9999 or greater, it is possible to correctly deliver a data packet".

Time bounded reachability: The system can reach a certain set of states within a certain time deadline with a given probability. For example, "with probability 0.975 or greater, it is possible to correctly deliver a data packet within 5 time units".

Invariance: The system does not leave a certain set of states with a given probability. For example, "with probability 0.875 or greater, the system never aborts".

Bounded response: The system inevitably reaches a certain set of states within a certain time deadline with a given probability. For example, "with probability 0.99 or greater, a data packet will always be delivered within 5 time units".

In Sections 5 and 7, we will see how each of these properties may be expressed in formalisms for which probabilistic timed automaton verification is possible.

4. Probabilistic timed structures

In this section, we introduce an underlying model for probabilistic timed automata, called *probabilistic timed structures*, which are obtained by augmenting the timed structures of [20] with a probabilistic choice over transitions, and take the form of a variant of Markov decision processes. More precisely, instead of a nondeterministic choice over transitions that consist of a real-valued duration and a next state, as is the case in traditional timed structures, the transition function of probabilistic timed structures results in a choice over pairs consisting of a duration and a *discrete probability distribution* over next states.

Definition 3 (*Probabilistic timed structure*). A *probabilistic timed structure* $\mathscr{M}$ is a labelled Markov decision process $(Q, Steps, L)$ where Q is a set of states, $Steps: Q \to 2^{\mathbb{R} \times \mu(Q)}$ is a function which assigns to each state $q \in Q$ a set $Steps(q)$ of pairs of the form (t, p) where $t \in \mathbb{R}$ and $p \in \mu(Q)$, and $L: Q \to 2^{\mathrm{AP}}$ is a state labelling function.

$Steps(q)$ is the set of transitions that can be nondeterministically chosen in state q. Each transition takes the form (t, p), where t represents the duration of the transition and p is the probability distribution used over the set of successor states. Therefore, given the nondeterministic choice of $(t, p) \in Steps(q)$ in state q, then, after t time units have elapsed, a probabilistic transition is made to state q' with probability $p(q')$.

Paths in a probabilistic timed structure arise by resolving both the nondeterministic and probabilistic choices. A *path* of the probabilistic timed structure $\mathscr{M} = (Q, Steps, L)$ is a nonempty finite or infinite sequence:

$$\omega = q_0 \xrightarrow{t_0, p_0} q_1 \xrightarrow{t_1, p_1} q_2 \xrightarrow{t_2, p_2} \cdots,$$

where $q_i \in Q$, $(t_i, p_i) \in Steps(q_i)$ and $p_i(q_{i+1}) > 0$ for all $0 \leqslant i \leqslant |\omega|$.

Sets of labelled paths are denoted in the following way. $Path_{\mathrm{fin}}$ is the set of finite paths, and $Path_{\mathrm{fin}}(q)$ is the set of paths in $Path_{\mathrm{fin}}$ such that $\omega(0) = q$. $Path_{\mathrm{ful}}$ is the set of infinite paths and $Path_{\mathrm{ful}}(q)$ is the set of paths in $Path_{\mathrm{ful}}$ such that $\omega(0) = q$.

Consider an infinite path ω of $\mathscr{M}$. A *position* of ω is a pair (i, t'), where $i \in \mathbb{N}$ and $t' \in \mathbb{R}$ such that $0 \leqslant t' \leqslant t_i$. The *state at position* (i, t'), denoted by $q_i + t'$. Given a path ω, $i, j \in \mathbb{N}$ and $t, t' \in \mathbb{R}$ such that $i \leqslant |\omega|$, $t \leqslant t_i$ and $t' \leqslant t_j$, then we say that the

position (j,t') *precedes* the position (i,t), written $(j,t') \prec (i,t)$, iff $j<i$, or $j=i$ and $t'<t$.

Definition 4 (*Duration of a path*). For any path ω of a probabilistic timed structure $\mathscr{M}$ and $0 \leqslant i \leqslant |\omega|$ we define $\mathscr{D}_\omega(i)$, the *elapsed time* until the ith transition, as follows: $\mathscr{D}_\omega(0)=0$ and for any $1 \leqslant i \leqslant |\omega|$:

$$\mathscr{D}_\omega(i) = \sum_{j=0}^{i-1} t_j.$$

Furthermore, an infinite path ω is *divergent* if for any $t \in \mathbb{R}$, there exists $j \in \mathbb{N}$ such that $\mathscr{D}_\omega(j) > t$.

We now introduce *adversaries* of probabilistic timed structures as functions which resolve all of the nondeterministic choices of the model.

Definition 5 (*Adversary of a probabilistic timed structure*). An *adversary* (or scheduler) of a probabilistic timed structure $\mathscr{M}=(Q,Steps,L)$ is a function A mapping every finite path ω of $\mathscr{M}$ to a pair (t,p) such that $A(\omega) \in Steps(last(\omega))$. Let $\mathscr{A}$ be the set of all adversaries of $\mathscr{M}$.

For an adversary A of a probabilistic timed structure $\mathscr{M}=(Q,Steps,L)$ we define $Path^A_{\text{fin}}$ to be the set of finite paths such that $step(\omega,i)=A(\omega^{(i)})$ for all $1 \leqslant i \leqslant |\omega|$, and $Path^A_{\text{ful}}$ to be the set of paths in $Path_{\text{ful}}$ such that $step(\omega,i)=A(\omega^{(i)})$ for all $i \in \mathbb{N}$.

With each adversary we associate a sequential Markov chain, which can be viewed as a set of paths in $\mathscr{M}$. Formally, if A is an adversary of the probabilistic timed structure $\mathscr{M}$, then $MC^A=(Path^A_{\text{fin}}, \mathbf{P}^A)$ is a Markov chain where

$$\mathbf{P}^A(\omega,\omega') = \begin{cases} p(q) & \text{if } A(\omega)=(t,p) \text{ and } \omega' = \omega \xrightarrow{t,p} q, \\ 0 & \text{otherwise.} \end{cases}$$

For any probabilistic timed structure and adversary A, let $\mathscr{F}^A_{Path}$ be the smallest σ-algebra on $Path^A_{\text{ful}}$ which contains the sets

$$\{\omega \mid \omega \in Path^A_{\text{ful}} \text{ and } \omega' \text{ is a prefix of } \omega\}$$

for all $\omega' \in Path^A_{\text{fin}}$.

We now define a measure $Prob^A$ on the σ-algebra $\mathscr{F}^A_{Path}$, by first defining the following function on the set of finite paths $Path^A_{\text{fin}}$.

Definition 6. Let A be an adversary of the probabilistic timed structure $\mathscr{M}$. Let $Prob^A_{\text{fin}} : Path^A_{\text{fin}} \to [0,1]$ be the mapping inductively defined on the length of paths in $Path^A_{\text{fin}}$ as follows. If $|\omega|=0$, then $Prob^A_{\text{fin}}(\omega)=1$.

Let $\omega' \in Path^A_{\text{fin}}$ be a finite path of A. If $\omega' = \omega \xrightarrow{t,p} q$ for some $\omega \in Path^A_{\text{fin}}$, then we let

$$Prob^A_{\text{fin}}(\omega') = Prob^A_{\text{fin}}(\omega) \cdot \mathbf{P}^A(\omega, \omega').$$

Definition 7. The measure $Prob^A$ on $\mathscr{F}^A_{Path}$ is the unique measure such that

$$Prob^A\{\omega \mid \omega \in Path^A_{\text{ful}} \text{ and } \omega' \text{ is a prefix of } \omega\} = Prob^A_{\text{fin}}(\omega').$$

When clear from the context, we drop the superscript denoting the adversary from the function $Prob^A_{\text{fin}}$ and the measure $Prob^A$.

A common restriction imposed in the study of real-time systems is that of *time divergence*. This requires that paths of a real-time system which are not divergent in the manner of Definition 4 are disregarded during analysis, because they exhibit unrealisable behaviour in which time is not allowed to pass beyond some bound. We now introduce the class of *divergent adversaries*, which resolve nondeterminism in such a way as to result in divergent behaviour with probability 1. Our aim is to disregard adversaries that are not divergent, as they exhibit unrealisable behaviour with positive probability.

Definition 8 (*Divergent adversary*). An adversary A of a probabilistic timed structure $(Q,Steps,L)$ is *divergent* if and only if

$$Prob\{\omega \mid \omega \in Path^A_{\text{ful}} \text{ and } \omega \text{ is divergent}\} = 1.$$

Let $\mathscr{A}_{div}$ be the set of all divergent adversaries.

Note that we only consider such *probabilistic* time divergent adversaries as opposed to a stronger definition in which an adversary is divergent if *all* of its corresponding paths are divergent. We motivate this choice in the next section.

4.1. Obtaining a probabilistic timed structure from a probabilistic timed automaton

This section will now show that the behaviour of a probabilistic timed automaton G can be stated formally in terms of a probabilistic timed structure $\mathscr{M}_G$. This structure consists of three components: the set of admissible states of G (those that satisfy the invariant condition); a transition relation that is both nondeterministic and probabilistic in nature, and includes time passage information; and a labelling function, which is derived from the labelling function $\mathscr{L}$ of G.

For convenience, we categorise distributions of $\mathscr{M}_G$ by assigning them a *type*. That is, if a distribution over states $\tilde{p}$ of $\mathscr{M}_G$ is derived from a particular distribution $p \in \bigcup_{s \in \mathscr{S}} prob(s)$ of G, then we say that $\tilde{p}$ has type p. We abuse notation to denote this by $\mathsf{type}(\tilde{p}) = p$. As we wish to allow for the fact that G may let time elapse without necessarily performing a subsequent discrete transition, we use the special symbol $\perp$ to denote the type of distribution of $\mathscr{M}_G$ that is induced by the passage of time

only (that is, without a subsequent discrete transition made according to a distribution of G).

Definition 9. For any probabilistic timed automaton G, define the probabilistic timed structure $\mathscr{M}_G = (Q_G, Steps_G, L_G)$ as follows:

States: A *state* of $\mathscr{M}_G$ is a pair $\langle s, v\rangle$, where $s \in \mathscr{S}$ is a node and $v \in \mathbb{R}^{\mathscr{X}}$ is a valuation such that v satisfies $inv(s)$. Let Q_G be the set of states of $\mathscr{M}_G$.

Transitions: The function $Steps_G : Q_G \to 2^{\mathbb{R}\times\mu(Q_G)}$ assigns to each state in Q_G a set of transitions, each of which take the form of a pair $(t, \tilde{p})$, comprising of a *time duration* $t \in \mathbb{R}$ and a *probability distribution* $\tilde{p} \in \mu(Q_G)$ over the set of states Q_G. Transitions are defined in two ways. For every $\langle s, v\rangle \in Q_G$:

(1) Let $(t, \tilde{p}) \in Steps_G\langle s, v\rangle$ if there exists $p \in prob(s)$ such that
   (a) the valuation $v + t$ satisfies $\tau_s(p)$;
   (b) the valuation $v + t'$ satisfies the invariant condition $inv(s)$ for all $0 \leqslant t' \leqslant t$; and
   (c) for any $\langle s', v'\rangle \in Q_G$:

$$\tilde{p}\langle s', v'\rangle = \sum_{\substack{X\subseteq\mathscr{X}\,\&\\ (v+t)[X:=0]=v'}} p(s', X).$$

   We refer to $\tilde{p}$ as having type p; that is, $\mathsf{type}(\tilde{p}) = p$.

(2) Let $(t, \tilde{p}) \in Steps_G\langle s, v\rangle$ if
   (a) the valuation $v + t'$ satisfies $inv(s)$ for all $0 \leqslant t' \leqslant t$; and
   (b) for any $\langle s', v'\rangle \in Q_G$:

$$\tilde{p}\langle s', v'\rangle = \begin{cases} 1 & \text{if } \langle s', v'\rangle = \langle s, v + t\rangle, \\ 0 & \text{otherwise.} \end{cases}$$

   We refer to $\tilde{p}$ as having type $\perp$; that is, $\mathsf{type}(\tilde{p}) = \perp$.

Labelling function: The labelling function $L : Q \to 2^{\mathrm{AP}}$ is defined as follows. For each $\langle s, v\rangle \in Q_G$, let $L_G\langle s, v\rangle = \mathscr{L}(s)$.

It is now possible to define the set $\mathscr{A}^G$ of adversaries of $\mathscr{M}_G$ using Definition 5. Note that $\mathscr{A}^G_{div}$ denotes the divergent adversaries of G. Where G is clear from the context, we drop the superscript from $\mathscr{A}^G$ and $\mathscr{A}^G_{div}$.

The following example motivates our choice of probabilistic time divergent adversaries, as given in Definition 8. Consider the probabilistic timed automaton G_2 in Fig. 2. If we consider any adversary of the corresponding probabilistic timed structure $\mathscr{M}_{G_2}$, then there will always exist a unique infinite path of the adversary which loops continually in s_1; that is, the adversary exhibits a path of the form

$$\omega = \langle s_1, v_0\rangle \xrightarrow{t_0,p} \langle s_1, v_1\rangle \xrightarrow{t_1,p} \langle s_1, v_2\rangle \xrightarrow{t_2,p} \cdots$$

such that $\sum_{i=0}^{\infty} t_i \leqslant 1$. It is clear that ω is not time divergent. Therefore, using a stronger definition of divergent adversaries which requires *all* of the paths of the

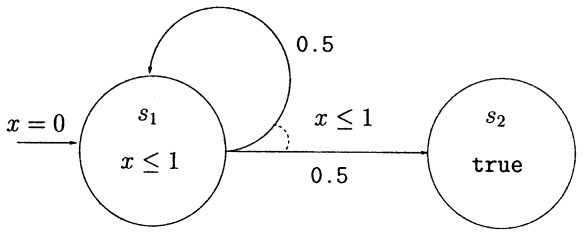

Fig. 2. The probabilistic timed automaton G_2.

adversary to be divergent, there would not exist *any* divergent adversaries of this model. However, the probability of the above path ω occurring is 0. Therefore, Definition 8 is motivated by the view that adversaries for which there exist unrealisable, non-divergent paths *should* be considered, provided that the probability measure of these paths is 0.

5. Probabilistic timed computation tree logic

We now describe the probabilistic real-time logic probabilistic timed computation tree logic (PTCTL) which can be used to specify properties of probabilistic timed systems. PTCTL synthesizes elements from two extensions of the branching temporal logic CTL, namely the real-time temporal logic TCTL [21] and the essentially equivalent, probabilistic temporal logics pCTL and PBTL [8, 6]. In particular, the temporal operator $\mathscr{U}$ ("until") and the path quantifiers $\forall$ and $\exists$ ("for all" and "there exists", respectively) are taken from CTL, the reset quantifier $z.\phi$ and the facility to refer directly to clock values are taken from TCTL, and the probabilistic operators $[\phi_1\ \exists\mathscr{U}\ \phi_2]_{\sqsupseteq\lambda}$ and $[\phi_1\ \forall\mathscr{U}\ \phi_2]_{\sqsupseteq\lambda}$ are taken from PBTL. Note that the reset quantifier $z.\phi$ is used to reset the clock z, so that ϕ is evaluated from a state at which $z=0$. Using our new logic, we can express properties such as, "with probability 0.6 or greater, the value of the system clock x does not exceed 3 before 5 time units have elapsed", which is represented as the PTCTL formula $z.[(x \leqslant 3)\ \forall\mathscr{U}\ (z=5)]_{\geqslant 0.6}$.

Similarly, we can write the properties of Section 3.1 in terms of PTCTL formulae. The property, "with probability 0.9999 or greater, it is possible to correctly deliver a data packet", can be written as $[\texttt{true}\ \exists\mathscr{U}\ ri]_{\geqslant 0.9999}$. The time bounded reachability property "with probability 0.975 or greater, it is possible to correctly deliver a data packet within 5 time units", can be expressed by the formula $z.[\texttt{true}\ \exists\mathscr{U}\ (ri \wedge z \leqslant 5)]_{\leqslant 0.975}$. The PTCTL formula $\neg[\texttt{true}\exists\mathscr{U} aa]_{>0.125}$ represents the property "with probability 0.875 or greater, the system never aborts" (that is, it is not true that the abort state *aa* can be reached with probability greater than 0.125). The bounded response property, "with probability 0.99 or greater, a data packet will always be delivered within 5 time units", can be written as $z.[\texttt{true}\ \forall\mathscr{U}\ (ri \wedge z \leqslant 5)]_{\geqslant 0.99}$.

As with TCTL, PTCTL employs a set of clock variables in order to express timing properties; for this purpose, we introduce a set of *formula clocks*, $\mathscr{Z}$, which is disjoint

from $\mathscr{X}$. Such clocks are assigned values by a *formula clock valuation* $\mathscr{E}: \mathscr{Z} \to \mathbb{R}$, which uses the notation for clock valuations in the standard way.

Definition 10 (*Syntax of PTCTL*). The syntax of *PTCTL* is defined as follows:

$$\phi ::= \mathtt{true} \mid a \mid \zeta \mid \phi \wedge \phi \mid \neg\phi \mid z.\phi \mid [\phi\ \exists\mathscr{U}\ \phi]_{\sqsupseteq\lambda} \mid [\phi\ \forall\mathscr{U}\ \phi]_{\sqsupseteq\lambda}$$

where $a \in AP$ is an *atomic proposition*, $\zeta \in \mathbf{Z}_{\mathscr{X}\cup\mathscr{Z}}$ is a *zone*, $z \in \mathscr{Z}$, $\lambda \in [0,1]$, and $\sqsupseteq$ is either $\geqslant$ or $>$.

Note that the values of system clocks in $\mathscr{X}$ and formula clocks in $\mathscr{Z}$ can be obtained from a state and a formula clock valuation, respectively. Take a particular $\zeta \in \mathbf{Z}_{\mathscr{X}\cup\mathscr{Z}}$, and consider the syntactic interpretation of ζ as a conjunction of constraints. Then, given a state $\langle s, v\rangle$ and a formula clock valuation $\mathscr{E}$, we denote by $\zeta[\langle s, v\rangle, \mathscr{E}]$ the boolean value obtained by replacing each occurrence of a system clock $x \in \mathscr{X}$ in ζ by $v(x)$, and each occurrence of a formula clock $z \in \mathscr{Z}$ in ζ by $\mathscr{E}(z)$.

As in the case of probabilistic timed automata, it is useful to identify the maximal integer constant that is referred to in a zone appearing in a PTCTL formula ϕ. Let $\mathbf{Z}_{\mathscr{X}\cup\mathscr{Z}}(\phi) = \{\zeta \in \mathbf{Z}_{\mathscr{X}\cup\mathscr{Z}} \mid \zeta \text{ is a subformula of } \phi\}$ be the set of zones occurring in ϕ, and define $c_{\max}(\phi) = \max\{c_{\max}(\zeta) \mid \zeta \in \mathbf{Z}_{\mathscr{X}\cup\mathscr{Z}}(\phi)\}$ be the required maximal constant.

Definition 11 (*Satisfaction relation for PTCTL*). Given a probabilistic timed structure $\mathscr{M} = (Q, Steps, L)$ and a set $\mathscr{A}$ of adversaries of $\mathscr{M}$, then for any state q of $\mathscr{M}$, formula clock valuation $\mathscr{E}$, and *PTCTL* formula ϕ, the satisfaction relation $q, \mathscr{E} \models_{\mathscr{A}} \phi$ is defined inductively as follows:

$q, \mathscr{E} \models_{\mathscr{A}} \mathtt{true}$ for all q and $\mathscr{E}$

$q, \mathscr{E} \models_{\mathscr{A}} a \Leftrightarrow a \in L(q)$

$q, \mathscr{E} \models_{\mathscr{A}} \zeta \Leftrightarrow \zeta[q, \mathscr{E}] = \mathtt{true}$

$q, \mathscr{E} \models_{\mathscr{A}} \phi_1 \wedge \phi_2 \Leftrightarrow q, \mathscr{E} \models_{\mathscr{A}} \phi_1$ and $q, \mathscr{E} \models_{\mathscr{A}} \phi_2$

$q, \mathscr{E} \models_{\mathscr{A}} \neg\phi \Leftrightarrow q, \mathscr{E} \not\models_{\mathscr{A}} \phi$

$q, \mathscr{E} \models_{\mathscr{A}} z.\phi \Leftrightarrow q, \mathscr{E}[z := 0] \models_{\mathscr{A}} \phi$

$q, \mathscr{E} \models_{\mathscr{A}} [\phi_1\ \exists\mathscr{U}\ \phi_2]_{\sqsupseteq\lambda} \Leftrightarrow Prob(\{\omega \mid \omega \in Path^A_{\mathrm{full}}(q)\ \&\ \omega, \mathscr{E} \models_{\mathscr{A}} \phi_1\ \mathscr{U}\ \phi_2\}) \sqsupseteq \lambda$ for some $A \in \mathscr{A}$

$q, \mathscr{E} \models_{\mathscr{A}} [\phi_1\ \forall\mathscr{U}\ \phi_2]_{\sqsupseteq\lambda} \Leftrightarrow Prob(\{\omega \mid \omega \in Path^A_{\mathrm{full}}(q)\ \&\ \omega, \mathscr{E} \models_{\mathscr{A}} \phi_1\ \mathscr{U}\ \phi_2\}) \sqsupseteq \lambda$ for all $A \in \mathscr{A}$

$\omega, \mathscr{E} \models_{\mathscr{A}} \phi_1\ \mathscr{U}\ \phi_2 \Leftrightarrow$ there exists a position (i,t) of ω such that $\omega(i) + t, \mathscr{E} + \mathscr{D}_\omega(i) + t \models_{\mathscr{A}} \phi_2$, and for all positions (j,t') of ω such that $(j,t') \prec (i,t), \omega(j) + t', \mathscr{E} + \mathscr{D}_\omega(j)v + t' \models_{\mathscr{A}} \phi_1 \vee \phi_2$.

6. Model checking probabilistic timed automata against PTCTL properties

Note that, because all clocks are real valued, the state space of a probabilistic timed automaton is infinite. However, a fundamental result of [3] is that the space of clock valuations of a timed automaton can be partitioned into a finite set of zones called *clock regions*, each containing a finite or infinite number of valuations which, as noted by Alur et al. [2], satisfy the same TCTL formulae. Combination of this partitioning with the discrete transitions of a timed automaton induces a structure called a *region graph*, which can be used for model checking. This section will show that a similar construction can be used for model checking probabilistic timed automata against PTCTL formulae.

6.1. Equivalence of clock valuations

We first define an equivalence relation on the space of clock valuations, which can then be used to obtain a finite partitioning of this space. Consider the probabilistic timed automaton G, and let $c = c_{\max}(G)$. Definition 12 to Lemma 14 recall standard notions of equivalence of clock valuations [2, 3].

Definition 12. For any $t \in \mathbb{R}$, $\lfloor t \rfloor$ denotes the integral part of t. Then, for any $t, t' \in \mathbb{R}$, t and t' *agree on their integral parts* if and only if:
(1) $\lfloor t \rfloor = \lfloor t' \rfloor$, and
(2) both t and t' are integers or neither is an integer.

Definition 13 (*Clock equivalence*). The valuations $v, v' \in \mathbb{R}^{\mathscr{X}}$ are *clock equivalent*, denoted by $v \cong v'$, if and only if they satisfy the following conditions:
(1) $\forall x \in \mathscr{X}$, either $v(x)$ and $v'(x)$ agree on their integral parts, or both $v(x) > c$ and $v'(x) > c$, and
(2) $\forall x, x' \in \mathscr{X}$, either $v(x) - v(x')$ and $v'(x) - v'(x')$ agree on their integral parts, or both $v(x) - v(x') > c$ and $v'(x) - v'(x') > c$.

Lemma 14. *Let the valuations $v, v' \in \mathbb{R}^{\mathscr{X}}$ be such that $v \cong v'$. Then the following conditions hold*:
(1) $v[X := 0] \cong v'[X := 0]$ *for all* $X \subseteq \mathscr{X}$,
(2) *for any zone* $\zeta \in \mathbf{Z}_{\mathscr{X}}(G)$ *appearing in the description of* G, v *satisfies* ζ *if and only if* v' *satisfies* ζ.

Proof. The proof follows from the definition of $\cong$. □

Note that clock equivalence classes can be regarded as special types of zones (see [3]). Let $[v]$ denote the equivalence class of $\cong$ to which v belongs. We refer to elements such as $\langle s, [v] \rangle$ as *regions*. Observe that, for finite c, clock equivalence has a finite number of classes. As $c_{\max}(G)$ is finite, and $\mathscr{S}$ is a finite set, we conclude that the set of the regions of G is also finite.

Remark. Note that a smaller set of clock equivalence classes can be obtained if, for each clock $x \in \mathscr{X}$, we maintain a maximal constant c_x with which x is compared in a zone of G, and then use the resulting maxima in the definition of clock equivalence. Although this may result in fewer regions, and thus improve the efficiency of the model checking techniques presented in this paper, we do not not use this approach for reasons of convenience. All of the results of this paper continue to hold if this approach is taken.

We now extend the concept of clock equivalence to formula clocks. Let $(v, \mathscr{E}) : \mathscr{X} \cup \mathscr{Z} \to \mathbb{R}$ be the clock valuation that assigns a real value to each of the system and formula clocks, and let $\mathbb{R}^{\mathscr{X} \cup \mathscr{Z}}$ be the set of all such valuations for G. For a $(v, \mathscr{E}) \in \mathbb{R}^{\mathscr{X} \cup \mathscr{Z}}$, and $X \subseteq \mathscr{X} \cup \mathscr{Z}$, we use the notation $(v, \mathscr{E})[X := 0]$ in the usual way. For some $t \in \mathbb{R}$, $(v + t, \mathscr{E} + t)$ denotes the clock valuation for which all clocks x in $\mathscr{X} \cup \mathscr{Z}$ take the value $(v, \mathscr{E})(x) + t$.

The equivalence relation for such a valuation is defined with respect to a particular PTCTL formula ϕ. Let $\mathscr{E}'$ be the restriction of $\mathscr{E}$ over the clocks of $\mathscr{Z}$ that are referred to in ϕ. We can then extend the equivalence relation from $\cong$ to $\cong^*$ simply by taking $(v, \mathscr{E}')$ instead of v and $\mathscr{X} \cup \mathscr{Z}$ instead of $\mathscr{X}$, and requiring that $c = \max\{c_{\max}(G), c_{\max}(\phi)\}$; the definition of equivalence classes of the form $[v, \mathscr{E}']$ then follows in an obvious manner. Furthermore, Lemma 14 holds for $\cong^*$ (in particular, part (2) applies to all zones $\zeta \in \mathbf{Z}_{\mathscr{X}}(G) \cup \mathbf{Z}_{\mathscr{X} \cup \mathscr{Z}}(\phi)$ appearing either in G or ϕ). Because our construction of the equivalence classes will always be with respect to a particular ϕ, we henceforth write $\mathscr{E}$ for $\mathscr{E}'$. An element of the form $\langle s, [v, \mathscr{E}] \rangle$ is called an *augmented region*. As $c_{\max}(\phi)$ is finite, we conclude that the set of augmented regions is also finite.

Let α be an equivalence class of the form $[v, \mathscr{E}]$. Observing that α is a zone, we recall the definition of clock resets on zones to denote by $\alpha[X := 0]$ the equivalence class obtained from α by setting all of the clocks in X to 0.

6.2. The region graph

6.2.1. Definition of the region graph

We now define an edge relation over the augmented regions to obtain the *region graph*. The nonprobabilistic region construction of [2] results in a state-labelled transition system which can be model checked using well-established methods. However, in our case the region graph takes the form of a Markov decision process for which there exist model checking techniques for temporal logics with probability bounds [8, 6].

First, we require some preliminary definitions. We explain how a region may be thought of as satisfying a conjunction of clock constraints represented as a zone; then we categorize regions in a number of ways.

Definition 15 (*Satisfaction of clock constraints*). Let α be an equivalence class of the relation $\cong^*$ on $\mathbb{R}^{\mathscr{X} \cup \mathscr{Z}}$ and $\zeta \in \mathbf{Z}_{\mathscr{X}}(G) \cup \mathbf{Z}_{\mathscr{X} \cup \mathscr{Z}}(\phi)$ be a zone either appearing in the

description of G or as a subformula of ϕ. Then α *satisfies* ζ if and only if, for any $(v,\mathscr{E})\in\alpha$, the value of ζ after substituting each occurrence of $x\in\mathscr{X}$ with $v(x)$, and each occurrence of $z\in\mathscr{Z}$ with $\mathscr{E}(z)$, is `true`. (Note that the value of ζ will be the same for all $(v,\mathscr{E})\in\alpha$, by the extension of Lemma 14(2) to augmented regions.)

Definition 16 (*Categorization of regions*). Let α and β be distinct clock equivalence classes of $\mathbb{R}^{\mathscr{X}\cup\mathscr{Z}}$.

Successor class: The equivalence class β is said to be the *successor* of α if and only if, for each $(v,\mathscr{E})\in\alpha$, there exists a positive $t\in\mathbb{R}$ such that $(v+t,\mathscr{E}+t)\in\beta$, and $(v+t',\mathscr{E}+t')\in\alpha\cup\beta$ for all $t'\leqslant t$.

x-**zero class**: For a clock $x\in\mathscr{X}\cup\mathscr{Z}$, the equivalence class α is said to be x-*zero* if and only if, for each $(v,\mathscr{E})\in\alpha$, $(v,\mathscr{E})(x)=0$.

x-**unbounded class**: For a clock $x\in\mathscr{X}\cup\mathscr{Z}$, the equivalence class α is said to be x-*unbounded* if and only if, for each $(v,\mathscr{E})\in\alpha$, $(v,\mathscr{E})(x)>\max(c_{\max}(G),c_{\max}(\phi))$.

The successor relation can be extended to augmented regions in the following way: $\langle s',\beta\rangle$ is the *successor region* of $\langle s,\alpha\rangle$ if $s'=s$ and $\beta=succ(\alpha)$. Similarly, the region $\langle s,\alpha\rangle$ is x-zero (x-unbounded) if α is x-zero (x-unbounded).

We now define a region graph which captures both the probabilistic transitions in G and the movement to new regions due to the passage of time, and which takes the form of a Markov decision process. As with probabilistic timed structures, transitions are made according to a two-phase process: firstly, a nondeterministic choice of a probabilistic distribution is made, and then a transition to a state in the support of the distribution is executed probabilistically. Naturally, in contrast to probabilistic timed structures, the transition relation of the region graph abstracts from exact timing information. Note that we associate a notion of type with certain transitions of the region graph, and use the notation type as introduced in Section 4.1.

The labelling function of the region graph requires the introduction of additional atomic propositions, which are taken to represent the satisfaction of the zones appearing as subformulae of ϕ. More precisely, for every zone ζ appearing in the given PTCTL formula ϕ (that is, for every zone in the set $\mathbf{Z}_{\mathscr{X}\cup\mathscr{Z}}(\phi)$), we extend the set AP with the atomic proposition a_ζ. We denote the resulting set of atomic propositions by AP^*.

Definition 17 (*Region graph*). The *region graph* $R(G,\phi)$ is defined to be the Markov decision process $(V^*,Steps^*,L^*)$. The vertex set V^* is the set of augmented regions. The transition function $Steps^*:V^*\to\mathscr{P}_{\mathrm{fin}}(\mu(V^*))$ includes three classes of transitions.[2] For each augmented region $\langle s,\alpha\rangle\in V^*$:

[2] If the model includes the distributions p_s^{inv} then we need to add an extra condition in the definition of transitions.

Passage of time : If the invariant condition $inv(s)$ is satisfied by $succ(\alpha)$, then $p^{s,\alpha}_{succ} \in Steps^*\langle s,\alpha\rangle$ where for any $\langle s',\beta\rangle \in V^*$:

$$p^{s,\alpha}_{succ}\langle s',\beta\rangle = \begin{cases} 1 & \text{if } \langle s',\beta\rangle = \langle s, succ(\alpha)\rangle, \\ 0 & \text{otherwise.} \end{cases}$$

Let the type of $p^{s,\alpha}_{succ}$ be $\perp$; that is, $\mathsf{type}(p^{s,\alpha}_{succ}) = \perp$.

Discrete transitions of G: $p^{s,\alpha}_{p'} \in Steps^*\langle s,\alpha\rangle$ if there exists $p' \in prob(s)$ and α satisfies the enabling condition $\tau_s(p')$ such that for any $s' \in \mathscr{S}$ and equivalence class β:

$$p^{s,\alpha}_{p'}\langle s',\beta\rangle = \sum_{\substack{X \subseteq \mathscr{X} \,\&\\ \alpha[X:=0]=\beta}} p'(s',X).$$

Let the type of $p^{s,\alpha}_{p'}$ be p'; that is, $\mathsf{type}(p^{s,\alpha}_{p'}) = p'$.

Self loops: Let $p^{s,\alpha}_{\text{loop}} \in Steps^*\langle s,\alpha\rangle$, where for any $\langle s',\beta\rangle \in V^*$:

$$p^{s,\alpha}_{\text{loop}}\langle s',\beta\rangle = \begin{cases} 1 & \text{if } \langle s',\beta\rangle = \langle s,\alpha\rangle, \\ 0 & \text{otherwise.} \end{cases}$$

Let the type of $p^{s,\alpha}_{\text{loop}}$ be $\perp$; that is, $\mathsf{type}(p^{s,\alpha}_{\text{loop}}) = \perp$.

The labelling function $L^* : V^* \to 2^{\mathrm{AP}^*}$ is defined in the following way. For each augmented region $\langle s,\alpha\rangle \in V^*$, we let

$$L^*\langle s,[v,\mathscr{E}]\rangle = \mathscr{L}(s) \cup \{a_\zeta \mid [v,\mathscr{E}] \text{ satisfies } \zeta, \text{ for } \zeta \in \mathbf{Z}_{\mathscr{X}\cup\mathscr{Y}}(\phi)\}.$$

Self-loops are included purely for technical convenience, and could be removed in certain contexts (for example, in an implementation of the model checking method).

Definition 18 (*Path on the region graph*). Given an augmented region $\langle s,\alpha\rangle$, a $\langle s,\alpha\rangle$-path is a finite or infinite path of the form

$$\omega^* = \langle s_0,\alpha_0\rangle \overset{p^{s_0,\alpha_0}}{\to} \langle s_1,\alpha_1\rangle \overset{p^{s_1,\alpha_1}}{\to} \langle s_2,\alpha_2\rangle \overset{p^{s_2,\alpha_2}}{\to} \cdots,$$

where $\langle s_0,\alpha_0\rangle = \langle s,\alpha\rangle$, $s_i \in \mathscr{S}$, α_i is an equivalence class of $\cong^*$ on $\mathbb{R}^{\mathscr{X}\cup\mathscr{Y}}$ and $p^{s_i,\alpha_i} \in Steps^*\langle s_i,\alpha_i\rangle$ such that $p^{s_i,\alpha_i}\langle s_{i+1},\alpha_{i+1}\rangle > 0$.

We define adversaries on the region graph $R(G,\phi)$ as follows:

Definition 19 (*Adversaries on the region graph*). An adversary A^* on the region graph is a function A^* mapping every finite path ω^* of $R(G,\phi)$ to a distribution p such that $p \in Steps^*(last(\omega^*))$.

We can then define the sets of paths $Path^*_{\text{fin}}$ and $Path^*_{\text{ful}}$, and those associated with an adversary, $Path^{A^*}_{\text{fin}}$ and $Path^{A^*}_{\text{ful}}$, as before. Note that an adversary on the region graph

$R(G,\phi)$ corresponds to an infinite number of adversaries on the underlying probabilistic timed structure. This is because the time component t in the choice of a transition $(t,\tilde{p})$ of $\mathscr{M}_G$ allows positive reals to be chosen, whereas the passage of time in any state $\langle s,\alpha\rangle$ of the region graph is given by the single distribution $p^{s,\alpha}_{succ}$.

With each adversary A^* we can associate a Markov chain. If A^* is an adversary of the region graph $R(G,\phi)$, then $MC^{A^*}=(Path^{A^*}_{\text{fin}},\mathbf{P}^{A^*})$ is a Markov chain where, for the augmented regions $\langle s,\alpha\rangle$, $\langle s',\alpha'\rangle$, and $last(\omega^*)=\langle s,\alpha\rangle$:

$$\mathbf{P}^{A^*}(\omega^*,\omega'^*)=\begin{cases} p^{s,\alpha}\langle s',\alpha'\rangle & \text{if } A^*(\omega^*)=p^{s,\alpha} \text{ and } \omega'^*=\omega^*\xrightarrow{p^{s,\alpha}}\langle s',\alpha'\rangle,\\ 0 & \text{otherwise.}\end{cases}$$

As in, for example, [6, 8], we define the function $Prob^{A^*}_{\text{fin}}$ on the set of finite paths $Path^{A^*}_{\text{fin}}$ and extend to the unique measure $Prob^{A^*}$ on the σ-algebra $\mathscr{F}^{A^*}_{Path}$.

6.2.2. Divergence on the region graph

We now introduce our notion of divergence on region graph paths, which is defined in terms of the classification of regions appearing infinitely often along such paths. The following definition is inspired by that of [9], and differs from that of [2] because, we allow for the possibility of an infinite number of discrete transitions in zero time. The definition proceeds by identifying two subsets of region graph paths. *Possibly progressive* paths are those which, for all clocks $x\in\mathscr{X}\cup\mathscr{Z}$, feature either infinitely many x-zero regions or, from some point onwards, the regions appearing along the path are x-unbounded. The intuition underlying the identification of such paths is that we are interested in characterizing behaviours in which, for each clock, either the clock is reset infinitely often, or eventually the value of the clock could be arbitrarily large. However, possibly progressive paths do not capture the concept of the divergence of time: consider a path for which there exists a clock $x\in\mathscr{X}\cup\mathscr{Z}$ such that, from some point onwards, all the regions along the path are x-zero. Then this paths is possibly progressive, but, because it has an infinite suffix for which all of the regions are x-zero, it only corresponds to the passage of a finite amount of time. We call such paths *zero*. Therefore, we say that a region graph path is *divergent* provided that it is possibly progressive but not zero.

Definition 20 (*Divergent region graph paths*). Let $\omega^*=\langle s_0,\alpha_0\rangle\xrightarrow{p^{s_0,\alpha_0}}\langle s_1,\alpha_1\rangle\xrightarrow{p^{s_1,\alpha_1}}\cdots$ be an infinite path of the region graph.

Possibly progressive: The path ω^* is *possibly progressive* if and only if, for each clock $x\in\mathscr{X}\cup\mathscr{Z}$, either:

(1) for every $i\in\mathbb{N}$, there exists $j\geqslant i$ such that α_j is an x-zero class, or
(2) there exists $i\in\mathbb{N}$ such that, for all $j\geqslant i$, α_j is an x-unbounded class.

Zero: The path ω^* is *zero* if and only if there exists a clock $x \in \mathscr{X}$ and $i \in \mathbb{N}$ such that, for all $j > i$, α_j is x-zero.

Divergent: The path ω^* is *divergent* if and only if it is possibly progressive and not zero.

Definition 21 (*Divergent adversaries on the region graph*). An adversary A^* is *divergent* if and only if

$$Prob^*\{\omega^* \mid \omega^* \in Path^{A^*}_{\text{ful}} \text{ and } \omega^* \text{ is divergent}\} = 1.$$

Let $\mathscr{A}^*_{div}$ be the set of divergent adversaries on the region graph.

Such divergent adversaries on the region graph $R(G,\phi)$ correspond to an infinite number of adversaries on the underlying probabilistic timed structure $\mathscr{M}_G$, some of which will be divergent in the sense of Definition 8. Conversely, for any divergent adversary of $\mathscr{M}_G$, the corresponding adversary on $R(G,\phi)$ is divergent.

6.3. Model checking for PTCTL using the region graph

A method for model checking probabilistic timed automata against PTCTL formulae will now be presented. This approach proceeds in three steps: construction of the region graph as a finite state representation of the probabilistic timed automaton in question; translating a formula of an extension of the probabilistic logic PBTL from the original PTCTL formula; and then resolving this new formula on the region graph.

First, we present an adjusted syntax of PBTL. Note that we omit PBTL's "bounded until" operator, because an equivalent, dense time, concept can be defined by nesting a PTCTL until operator within a reset quantifier, and its "next step" operator, which has no analogue in the case of dense real-time. However, we extend PBTL with a reset quantifier expression.

Definition 22 (*Syntax of PBTL*). The syntax of PBTL is defined as follows:

$$\Phi ::= \texttt{true} \mid a \mid \Phi \wedge \Phi \mid \neg\Phi \mid z.\Phi \mid [\Phi\ \exists\mathscr{U}\ \Phi]_{\sqsupseteq\lambda} \mid [\Phi\ \forall\mathscr{U}\ \Phi]_{\sqsupseteq\lambda}$$

where $a \in \mathrm{AP}$ is an atomic proposition, $z \in \mathscr{Z}$, $\lambda \in [0,1]$, and $\sqsupseteq$ is either $\geqslant$ or $>$.

Definition 23 (*Satisfaction relation for PBTL*). Given a region graph $R(G,\phi)$ and a set $\mathscr{A}^*$ of adversaries on $R(G,\phi)$, then for any augmented region $\langle s,[v,\mathscr{C}]\rangle$ of $R(G,\phi)$, and PBTL formula Φ, the satisfaction relation $\langle s,[v,\mathscr{C}]\rangle \models_{\mathscr{A}^*} \Phi$ is defined inductively

Subformula of ϕ_i	Subformula of Φ_i
`true`	`true`
a	a
ζ	a_ζ
$\phi_1 \wedge \phi_2$	$\Phi_1 \wedge \Phi_2$
$\neg\phi$	$\neg\Phi$
$z.\phi$	$z.\Phi$
$[\phi_1\ \exists\mathcal{U}\ \phi_2]_{\sqsupseteq\lambda}$	$[\Phi_1\ \exists\mathcal{U}\ \Phi_2]_{\sqsupseteq\lambda}$
$[\phi_1\ \forall\mathcal{U}\ \phi_2]_{\sqsupseteq\lambda}$	$[\Phi_1\ \forall\mathcal{U}\ \Phi_2]_{\sqsupseteq\lambda}$

Fig. 3. Rules for the derivation of a PBTL formula from a PTCTL formula.

as follows:

$$
\begin{aligned}
&\langle s,[v,\mathscr{E}]\rangle \models_{\mathscr{A}^*} \texttt{true} && \text{for all } \langle s,[v,\mathscr{E}]\rangle\\
&\langle s,[v,\mathscr{E}]\rangle \models_{\mathscr{A}^*} a && \Leftrightarrow a \in L^*\langle s,[v,\mathscr{E}]\rangle\\
&\langle s,[v,\mathscr{E}]\rangle \models_{\mathscr{A}^*} \Phi_1 \wedge \Phi_2 && \Leftrightarrow \langle s,[v,\mathscr{E}]\rangle \models_{\mathscr{A}^*} \Phi_1 \text{ and } \langle s,[v,\mathscr{E}]\rangle \models_{\mathscr{A}^*} \Phi_2\\
&\langle s,[v,\mathscr{E}]\rangle \models_{\mathscr{A}^*} \neg\Phi && \Leftrightarrow \langle s,[v,\mathscr{E}]\rangle \not\models_{\mathscr{A}^*} \Phi\\
&\langle s,[v,\mathscr{E}]\rangle \models_{\mathscr{A}^*} z.\Phi && \Leftrightarrow \langle s,[v,\mathscr{E}[z:=0]]\rangle \models_{\mathscr{A}^*} \Phi\\
&\langle s,[v,\mathscr{E}]\rangle \models_{\mathscr{A}^*} [\Phi_1\ \exists\mathcal{U}\ \Phi_2]_{\sqsupseteq\lambda} && \Leftrightarrow Prob^*(\{\omega \mid \omega \in Path^{A^*}_{ful}\langle s,[v,\mathscr{E}]\rangle\ \&\\
& && \quad \omega \models_{\mathscr{A}^*} \Phi_1\ \mathcal{U}\ \Phi_2\}) \sqsupseteq \lambda \quad \text{for some } A^* \in \mathscr{A}^*\\
&\langle s,[v,\mathscr{E}]\rangle \models_{\mathscr{A}^*} [\Phi_1\ \forall\mathcal{U}\ \Phi_2]_{\sqsupseteq\lambda} && \Leftrightarrow Prob^*(\{\omega \mid \omega \in Path^{A^*}_{ful}\langle s,[v,\mathscr{E}]\rangle\ \&\\
& && \quad \omega \models_{\mathscr{A}^*} \Phi_1\ \mathcal{U}\ \Phi_2\}) \sqsupseteq \lambda \quad \text{for all } A^* \in \mathscr{A}^*\\
&\omega \models_{\mathscr{A}^*} \Phi_1\ \mathcal{U}\ \Phi_2 && \Leftrightarrow \text{there exists } i \in \mathbb{N}, \text{ such that } \omega(i) \models_{\mathscr{A}^*} \Phi_2,\\
& && \quad \text{and for all } j \in \mathbb{N} \text{ such that } 0 \leqslant j < i \text{ and}\\
& && \quad \omega(j) \models_{\mathscr{A}^*} \Phi_1.
\end{aligned}
$$

For technical reasons, we require the following lemma, which follows immediately from the definition of the semantics of the formula $\Phi_1\ \mathcal{U}\ \Phi_2$ as given in Definition 23. Intuitively, the lemma gives a slightly different, but equivalent, semantics for the formula $\Phi_1\ \mathcal{U}\ \Phi_2$.

Lemma 24. *For any path ω of $(R(G,\phi),L^*)$, set of adversaries $\mathscr{A}^*$ on $R(G,\phi)$, and PBTL formulae Φ_1,Φ_2, we have $\omega \models_{\mathscr{A}^*} \Phi_1\ \mathcal{U}\ \Phi_2$ if and only if there exists $i \in \mathbb{N}$ such that $\omega(i) \models_{\mathscr{A}^*} \Phi_2$, and for all $j \in \mathbb{N}$ such that $0 \leqslant j \leqslant i$, $\omega(j) \models_{\mathscr{A}^*} \Phi_1 \vee \Phi_2$.*

Furthermore, a PBTL formula, Φ, can be *derived from* a PTCTL formula, ϕ, by applying the rules in Fig. 3 inductively.

Consider the probabilistic timed automaton given in Fig. 4. A region construction of the probabilistic timed automaton of G_2 is given in Fig. 5 (note that we omit self-loops of the form $p^{s,\alpha}_{\text{loop}}$ from the figure). As before, the probabilistic transitions are linked with an arc at their source vertex. In order for the reader to easily comprehend the

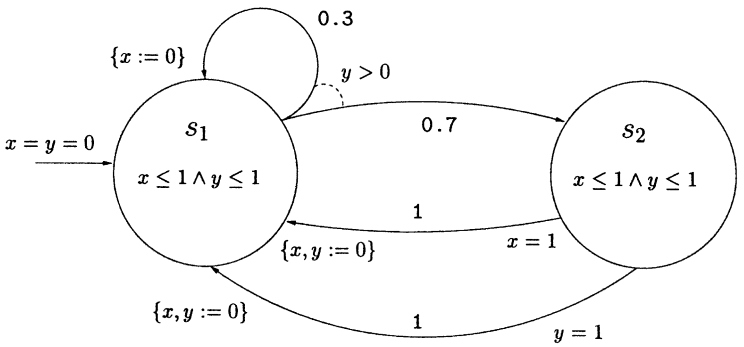

Fig. 4. The probabilistic timed automaton G_3.

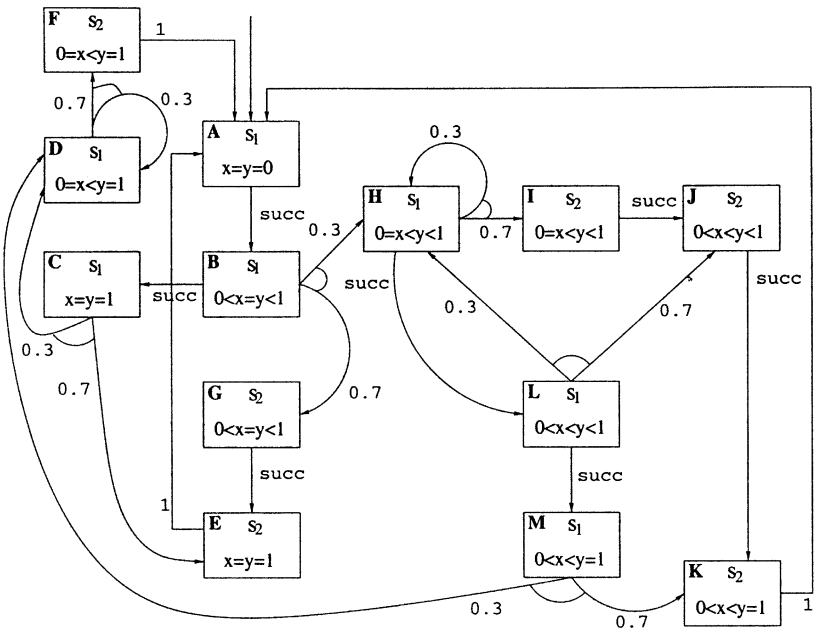

Fig. 5. The region graph of the probabilistic timed automaton G_3.

behaviour of the region graph, each vertex has been labelled with a constraint that is satisfied by all of the clock valuations within that augmented region. Consider the following PTCTL formula:

$$\phi_1 = [(y = 0)\ \exists\mathcal{U}\ (x > 0) \wedge [(x > 0)\ \exists\mathcal{U}\ (y = 0)]_{\geqslant 0.7}]_{\geqslant 1}.$$

ϕ_1 can be interpreted over this graph by first converting it into the equivalent PBTL formula:

$$\Phi_1 = [a_{(y=0)}\ \exists\mathcal{U}\ a_{(x>0)} \wedge [a_{(x>0)}\ \exists\mathcal{U}\ a_{(y=0)}]_{\geqslant 0.7}]_{\geqslant 1}.$$

Φ_1 is satisfied by this region graph, and therefore we conclude that the probabilistic timed automaton G_2 satisfies ϕ_1. Note that the following PTCTL formula, ϕ_2, is *not* satisfied by the region graph:

$$\phi_2 = [(y = 0)\ \exists\mathscr{U}\ (x > 0) \wedge [(x > 0)\ \exists\mathscr{U}\ (y = 0)]_{>0.7}]_{\geqslant 1}.$$

Proposition 25 (Correctness of the model checking procedure). *Given the probabilistic timed automaton G, we say that the state $\langle s, v\rangle$ of $\mathscr{M}_G$ and formula clock valuation $\mathscr{E}$ satisfy the PTCTL formula ϕ if and only if vertex $\langle s, [v, \mathscr{E}]\rangle$ of $R(G, \phi)$ satisfies the PBTL formula Φ, where Φ is derived from ϕ.*

Proof. We proceed to show this by induction on the structure of ϕ. The case of `true` and the boolean connectives, $\neg$ and $\wedge$, are self-evident.

If $\phi = a$, where $a \in \mathrm{AP}$, then it is true for state $\langle s, v\rangle$ of $\mathscr{M}_G$ and all formula clock valuations if and only if $a \in L\langle s, v\rangle$. We also know that $a \in L\langle s, v\rangle$ if and only if $a \in \mathscr{L}(s)$. By Definition 17, $a \in L^*\langle s, [v, \mathscr{E}]\rangle$ if $a \in \mathscr{L}(s)$, so $\Phi = a$ is true for the vertex $\langle s, [v, \mathscr{E}]\rangle$.

If $\phi = \zeta$, then the state $\langle s, v\rangle$ of $\mathscr{M}_G$ and formula clock valuation $\mathscr{E}$ satisfies ζ if $\zeta[\langle s, v\rangle, \mathscr{E}] = \mathtt{true}$. Then, from Definition 17, $a_\zeta \in L^*\langle s, [v, \mathscr{E}]\rangle$. Because $\Phi = a_\zeta$, and Φ is derived from ϕ, both ϕ and Φ resolve to true in $\langle s, v\rangle, \mathscr{E}$ and $\langle s, [v, \mathscr{E}]\rangle$, respectively.

If $\phi = z.\phi_1$, then, for a given state $\langle s, v\rangle$ and formula clock valuation $\mathscr{E}$ that satisfies ϕ, we know that the augmented region $\langle s, [v, \mathscr{E}]\rangle$ will also satisfy $z.\Phi_1$, by observing the following argument. By Definition 11,

$$\begin{aligned}
\langle s, v\rangle, \mathscr{E} \models_{\mathscr{A}^G_{div}} z.\phi &\Leftrightarrow \langle s, v\rangle, \mathscr{E}[z := 0] \models_{\mathscr{A}^G_{div}} \phi \\
&\Leftrightarrow \langle s, [v, \mathscr{E}[z := 0]]\rangle \models_{\mathscr{A}^*_{div}} \Phi \quad \text{by induction} \\
&\Leftrightarrow \langle s, [v, \mathscr{E}]\rangle \models_{\mathscr{A}^*_{div}} z.\Phi \quad \text{by Definition 23.}
\end{aligned}$$

Now we show that $\langle s, v\rangle, \mathscr{E} \models_{\mathscr{A}^G_{div}} [\phi_1\ \exists\mathscr{U}\ \phi_2]_{\sqsupseteq\lambda}$, if and only if $\langle s, [v, \mathscr{E}]\rangle \models_{\mathscr{A}^*_{div}} [\Phi_1\ \exists\mathscr{U}\ \Phi_2]_{\sqsupseteq\lambda}$. Our presentation is split into three sections:

1. Showing that, for a path ω of $\mathscr{M}_G$, a corresponding path of $R(G, \phi)$, $[\omega]$, can be constructed. Furthermore, ω is divergent if and only if $[\omega]$ is divergent. It also follows that, given path ω^* of the region graph, we can construct ω such that $[\omega] = \omega^*$.
2. Showing that the two finite paths ω and $[\omega]$ are associated with the same probability value.
3. Showing $\omega, \mathscr{E} \models_{\mathscr{A}^G_{div}} \phi_1\ \mathscr{U}\ \phi_2$ if and only if $[\omega] \models_{\mathscr{A}^*_{div}} \Phi_1\ \mathscr{U}\ \Phi_2$, where the initial augmented state of $[\omega]$ comprises of $\mathscr{E}$.

1. Consider the following property, which shall henceforth be referred to as the *sequence property*. Take a particular node of G, $s \in \mathscr{S}$, and a clock valuation $(v, \mathscr{E})$. We say the sequence of equivalence classes, $[v + d_1, \mathscr{E} + d_1], \ldots, [v + d_k, \mathscr{E} + d_k]$, has the sequence property if each equivalence class satisfies $inv(s)$, $d_i \in \mathbb{R}$ for all $1 \leqslant i \leqslant k$, and for all $1 \leqslant l < k$, $succ([v + d_l, \mathscr{E} + d_l]) = [v + d_{l+1}, \mathscr{E} + d_{l+1}]$. Then, for every time value

$d \in \mathbb{R}$, where $d_1 \leqslant d \leqslant d_k$, we know that $(v+d, \mathscr{E}+d) \cong^* (v+d_j, \mathscr{E}+d_j)$, for some $1 \leqslant j \leqslant k$. We can then write $(v+d, \mathscr{E}+d) \in [v+d_j, \mathscr{E}+d_j]$.

The sequence property allows us to state the following. Consider the path ω of $\mathscr{M}_G$, such that

$$\omega = \langle s_0, v_0 \rangle \overset{t_0, \tilde{p}_0}{\rightarrow} \langle s_1, v_1 \rangle \overset{t_1, \tilde{p}_1}{\rightarrow} \cdots.$$

Take a particular $i \geqslant 0$. From $\langle s_i, v_i \rangle$, and letting t_i time units elapse, we may pass through a number of equivalence classes before taking the discrete transition $\tilde{p}_i$. We let m_i be this number. Let $v_{i0} = \langle s_i, [v_i, \mathscr{E} + \mathscr{D}_\omega(i)] \rangle$, and, if $m_i > 0$, let $v_{ij} = \langle s_i, [v_i + d_j, \mathscr{E} + \mathscr{D}_\omega(i) + d_j] \rangle$ for all $1 \leqslant j \leqslant m_i$, and some $d_j \in \mathbb{R}$.

Furthermore, we emulate the choice of the distribution $\tilde{p}_i$ by the choice of $p_i^{v_{im_i}}$ in the vertex v_{im_i} so that $\mathsf{type}(\tilde{p}_i) = \mathsf{type}(p_i^{v_{im_i}})$. That is, we choose $p_i^{v_{im_i}}$ to be derived from a discrete transition of the same type as $\tilde{p}_i$, or a self loop if $\mathsf{type}(\tilde{p}_i) = \perp$. Then we can construct the finite path $[\omega_i]$ of the region graph, such that

$$[\omega_i] = v_{i0} \overset{v_{i0}}{\underset{p_{succ}}{\rightarrow}} v_{i1} \overset{v_{i1}}{\underset{p_{succ}}{\rightarrow}} \cdots v_{im_i} \overset{v_{im_i}}{\underset{p_i}{\rightarrow}} v_{(i+1)0}.$$

Note that $[\omega_i]$ must comprise of at least one transition. Let $[\omega] = [\omega_0][\omega_1]\cdots$ be the concatenation of all such segments.

This construction also works in the opposite direction. Let $\omega^* = \langle s_0, \alpha_0 \rangle \overset{p^{s_0,\alpha_0}}{\rightarrow} \langle s_1, \alpha_1 \rangle \overset{p^{s_1,\alpha_1}}{\rightarrow} \cdots$ be a path of the region graph. We proceed to construct the path ω of $\mathscr{M}_G$ inductively by progressing along the length of the path ω^*. For the base case of length 0, let $\langle s_0, v_0 \rangle$ be the state such that $v_0, \mathscr{E} \in \alpha_0$ for some formula clock valuation $\mathscr{E} \in \mathbb{R}^{\mathscr{Z}}$; then, $\langle s_0, v_0 \rangle$ is the first state of the path ω. Now say we have partially constructed ω up to length i, for some $i \in \mathbb{N}$. Let $\langle s_i, v_i \rangle = \omega(i)$. Then the transition $\langle s_i, \alpha_i \rangle \overset{p^{s_i,\alpha_i}}{\rightarrow} \langle s_{i+1}, \alpha_{i+1} \rangle$ of ω^* can be copied by a transition $\langle s_i, v_i \rangle \overset{t_i, \tilde{p}_i}{\rightarrow} \langle s_{i+1}, v_{i+1} \rangle$, where $t_i \in \mathbb{R}$, $\mathsf{type}(\tilde{p}_i) = \mathsf{type}(p^{s_i,\alpha_i})$, and $v_{i+1}, \mathscr{E} + \mathscr{D}_\omega(i+1) \in \alpha_{i+1}$. Then it follows that, given ω^* of $R(G, \phi)$, we can construct ω of $\mathscr{M}_G$ such that $[\omega] = \omega^*$.

We now show that, if the path ω of $\mathscr{M}_G$ is divergent, then $[\omega]$ will also be divergent, where $[\omega]$ is the region graph path constructed from ω. First, suppose for a contradiction that $[\omega]$ is a zero path, then by definition there exists $i \in \mathbb{N}$ and $x \in \mathscr{X} \cup \mathscr{Z}$ such that $[\omega](j)$ is x-zero for all $j > i$. By the construction of $[\omega]$ above, this corresponds to the situation in which the value of x does not advance from some point on in the path ω, and therefore time also cannot advance. However, this contradicts the fact that ω is divergent, and hence $[\omega]$ is not zero.

Secondly, for any clock $x \in \mathscr{X} \cup \mathscr{Z}$, if there exists $i \in \mathbb{N}$ such that there does not exist an x-zero region after the vertex $[\omega](i)$, then the clock x is not reset after this point in the path. Therefore, since the total time elapsed in ω must exceed any bound, the value of the clock x must also exceed any bound. Therefore, there must exist a $j \geqslant i$ for which all regions appearing along the path after $[\omega](j)$ are x-unbounded. We conclude that $[\omega]$ must be possibly progressive, and therefore $[\omega]$ is also be divergent.

Conversely, we can show that, if the region graph path ω^* is divergent, then the path ω of $\mathscr{M}_G$ that is constructed from ω^* can be chosen to be divergent. Firstly, as

ω^* is not a zero path, it follows that it is possible to select the states and transitions along ω such that it is possible for a positive amount of time to elapse infinitely often. Secondly, as ω^* is possibly progressive, it follows that the value of each clock is either reset infinitely often or is not bounded from above from some point along the path onwards. Therefore, the durations of time transitions are not forced to converge to 0 to prevent a clock from exceeding an upper bound. We conclude that the transitions along ω can be chosen in such a way as to ensure divergence.

2. Now, we show that the probability value associated with all of the finite prefixes of ω and $[\omega]$ are the same. Consider the segment of ω, $\omega_i = \langle s_i, v_i\rangle \overset{t_i,\tilde{p}_i}{\rightarrow} \langle s_{i+1}, v_{i+1}\rangle$, and the corresponding segment of $[\omega]$, constructed as in condition 1:

$$[\omega_i] = v_{i0} \overset{v_{i0}}{\underset{p_{succ}}{\rightarrow}} v_{i1} \overset{v_{i1}}{\underset{p_{succ}}{\rightarrow}} \cdots v_{im_i} \overset{v_{im_i}}{\underset{p_i}{\rightarrow}} v_{(i+1)0},$$

where $p_i^{v_{im_i}}$ is either a discrete transition or a self loop. We wish to show that $Prob_{\text{fin}}(\omega_i) = Prob^*_{\text{fin}}([\omega_i])$. Say $m_i \geqslant 1$, and consider the transition $v_{ij} \overset{v_{ij}}{\underset{p_{succ}}{\rightarrow}} v_{i(j+1)}$, for $1 \leqslant j < m_i$. Then, from the sequence property and the above construction of $[\omega]$, we know that $v_{i(j+1)}$ is a time successor of v_{ij}, and therefore $p_{succ}^{v_{ij}} = 1$. Therefore (and in the case in which $m_i = 0$), our problem reduces to showing that

$$Prob_{\text{fin}}(\langle s_i, v_i\rangle \overset{t_i,\tilde{p}_i}{\rightarrow} \langle s_{i+1}, v_{i+1}\rangle) = Prob^*_{\text{fin}}(v_{im_i} \overset{v_{im_i}}{\underset{p_i}{\rightarrow}} v_{(i+1)0}).$$

By the definitions of *Prob* and *Prob**, this reduces to showing that

$$\tilde{p}_i\langle s_{i+1}, v_{i+1}\rangle = p_i^{v_{im_i}}\langle s_i, [v_{i+1}, \mathscr{E} + \mathscr{D}_\omega(i+1)]\rangle.$$

We have two cases, depending on whether $\mathsf{type}(\tilde{p}_i) = \mathsf{type}(p_i^{v_{im_i}}) = \perp$ or $\mathsf{type}(\tilde{p}_i) = \mathsf{type}(p_i^{v_{im_i}}) \neq \perp$.

Case: $\mathsf{type}(\tilde{p}_i) = \mathsf{type}(p_i^{v_{im_i}}) \neq \perp$. Then $\mathsf{type}(\tilde{p}_i) = \mathsf{type}(p_i^{v_{im_i}}) = p_i$, for some distribution $p_i \in prob(s_i)$. Recall from Definition 9 that

$$\tilde{p}_i\langle s_{i+1}, v_{i+1}\rangle = \sum_{\substack{X \subseteq \mathscr{X} \,\&\\ (v_i + t_i)[X:=0] = v_{i+1}}} p_i(s_{i+1}, X)$$

and from the definition of the region graph:

$$p_i^{s_i,\alpha}\langle s_{i+1}, \beta\rangle = \sum_{\substack{X \subseteq \mathscr{X} \&\\ \alpha[X:=0] = \beta}} p_i(s_{i+1}, X),$$

where $\alpha = [v_i + d_{m_i}, \mathscr{E} + \mathscr{D}_\omega(i) + d_{m_i}]$ and $\beta = [v_{i+1}, \mathscr{E} + \mathscr{D}_\omega(i+1) + d_{i+1}]$. We know that, for any $X \subseteq \mathscr{X}$, $(v_i + t_i)[X := 0] \in [v_i + d_{m_i}][X := 0]$, and, trivially, that $v_{i+1} \in [v_{i+1}]$, and so the combinations of $X \subseteq \mathscr{X}$ used in both summations above will be the same.

Therefore, the same probability values will be summed in the case of $\mathscr{M}_G$ and that of $R(G,\phi)$, and we can conclude that $\tilde{p}_i\langle s_{i+1}, v_{i+1}\rangle = p_i^{s_i,\alpha}\langle s_{i+1}, \beta\rangle$.

Case: $\mathsf{type}(\tilde{p}_i) = \mathsf{type}(p_i^{v_{im_i}}) = \perp$. Observe that, as $p_i^{v_{im_i}}$ is not a time successor transition, then it must be a self-loop. Then, from the definitions of transitions with type $\perp$ in Definition 9 and self-loops in Definition 17:

$$\tilde{p}_i\langle s_{i+1}, v_{i+1}\rangle = p_i^{v_{im_i}}\langle s_i, [v_{i+1}, \mathscr{E} + \mathscr{D}_\omega(i+1)]\rangle = 1.$$

We can repeat such a process for all $i \in \mathbb{N}$ and, by the definitions of $Prob_{\text{fin}}$ and $Prob^*_{\text{fin}}$, show that the probability value associated with all of the finite prefixes of ω and $[\omega]$ are the same.

3. Next, we prove $\omega, \mathscr{E} \models_{\mathscr{A}^G_{div}} \phi_1 \ \mathscr{U} \ \phi_2$ if and only if $[\omega] \models_{\mathscr{A}^*_{div}} \Phi_1 \ \mathscr{U} \ \Phi_2$. If $\omega(i) = \langle s_i, v_i\rangle$ for all $i \in \mathbb{N}$, then $\omega, \mathscr{E} \models_{\mathscr{A}^G_{div}} \phi_1 \ \mathscr{U} \ \phi_2$

$\Leftrightarrow$ exists position (i,t) of ω such that $\omega(i) + t, \mathscr{E} + \mathscr{D}_\omega(i) + t \models_{\mathscr{A}^G_{div}} \phi_2$
and for all positions (j,t') of ω such that $(j,t') \prec (i,t)$,
$\omega(j) + t', \mathscr{E} + \mathscr{D}_\omega(j) + t' \models_{\mathscr{A}^G_{div}} \phi_1 \vee \phi_2$ by Definition 11

$\Leftrightarrow$ exists position (i,t) of ω such that $\langle s_i, [v_i + t, \mathscr{E} + \mathscr{D}_\omega(i) + t]\rangle \models_{\mathscr{A}^*_{div}} \Phi_2$
and for all positions (j,t') of ω such that $(j,t') \prec (i,t)$,
$\langle s_j, [v_j + t', \mathscr{E} + \mathscr{D}_\omega(j) + t']\rangle \models_{\mathscr{A}^*_{div}} \Phi_1 \vee \Phi_2$ by induction

$\Leftrightarrow$ $\exists i' \in \mathbb{N}$ such that $[\omega](i') \models_{\mathscr{A}^*_{div}} \Phi_2$ and $[\omega](j') \models_{\mathscr{A}^*_{div}} \Phi_1 \vee \Phi_2 \forall j' \leqslant i'$
by construction of $[\omega]$

$\Leftrightarrow$ $[\omega] \models_{\mathscr{A}^*_{div}} \Phi_1 \ \mathscr{U} \ \Phi_2$ by Lemma 24.

It follows by the definition of adversaries, both on probabilistic timed structures and the region graph, and the construction in condition 1, that for all $A \in \mathscr{A}^G_{div}$, there exists an adversary $[A] \in \mathscr{A}^*_{div}$ such that, for some $\mathscr{E}$,

$$Path^{[A]}_{\text{ful}}\langle s, [v, \mathscr{E}]\rangle = \{[\omega] \mid \omega \in Path^A_{\text{ful}}\langle s, v\rangle\}.$$

Similarly, we can show that, for all adversaries $A^* \in \mathscr{A}^*_{div}$ of the region graph, there exists an adversary $A \in \mathscr{A}^G_{div}$ such that $[A] = A^*$.

From condition 2, we know that the probability values associated with all finite prefixes of ω and $[\omega]$ are the same. Then we can conclude that

$$\begin{aligned} &Prob^*\{\omega^* \mid \omega^* \in Path^{A^*}_{\text{ful}}\langle s, [v, \mathscr{E}]\rangle \ \&\ \omega \models_{\mathscr{A}^*_{div}} \Phi_1 \ \mathscr{U} \ \Phi_2\} \\ &\quad = Prob\{\omega \mid \omega \in Path^A_{\text{ful}}\langle s, v\rangle \ \&\ \omega, \mathscr{E} \models_{\mathscr{A}^G_{div}} \phi_1 \ \mathscr{U} \ \phi_2\} \end{aligned}$$

for some $A \in \mathscr{A}^G_{div}$. $\square$

Using the transformation presented above, we can obtain a PBTL formula, Φ, from the PTCTL formula, ϕ. Now, we can use the model checking algorithm of [6] in order to verify whether the PBTL formula Φ holds in an initial state of the region graph, $\langle \bar{s}, (\mathbf{0}, \mathscr{E})\rangle$, where, for all $x \in \mathscr{X}$, $(\mathbf{0}, \mathscr{E})(x) = 0$ and $\mathscr{E}$ is an arbitrary formula clock valuation.

7. Model checking probabilistic timed automata against reachability properties

Although the verification technique of the previous section can establish the correctness of a probabilistic timed automaton model against a broad class of properties, it suffers from potential inefficiency as a result of the high granularity involved in the region construction of the finite state space. In particular, the size of the region graph is exponential in the number of clocks and the magnitude of the maximal constants with which they are compared in zones, either in the system description or in the temporal logic formula. Consideration of a narrower class of properties than those of PTCTL allows us to adopt a different approach when constructing a finite-state representation of a probabilistic timed automaton.

Here, we extend the real-time reachability properties of [13, 24] with probability to obtain *probabilistic real-time reachability properties*. Such properties are expressed in terms of a target set of states and a probability bound; for example, the property "with probability 0.9999 or greater, it is possible to correctly deliver a data packet" from the example in Section 3.1 can be expressed as a probabilistic real-time reachability property. It will also be shown how to represent invariance properties, such as "with probability 0.875 or greater, the system never aborts", and time bounded reachability properties, such as "with probability 0.975 or greater, it is possible to correctly deliver a data packet within 5 time units", as reachability properties.

As in the nonprobabilistic, real-time reachability case, our finite-state model derived from the probabilistic timed automaton G is obtained, not by the region construction, but by *forward* search through the infinite state space of G. Once this has been done, *probabilistic reachability* analysis is then performed on the finite state model through computation of probabilities using linear programming [15, 16]. We introduce the reachability algorithm in Sections 7.1 and 7.2, and, in Section 7.3, give an example of its application to the communication protocol of Section 3.1. In Section 7.4, it will be shown that the maximal probability of reaching the target set of states computed for the finite state representation is an *upper bound* on the actual maximal probability of the probabilistic timed automaton G reaching the target set. We conclude that our technique is less appropriate for the verification of reachability properties, for which we wish to establish whether the maximal probability of reaching a target set of states exceeds some bound, than invariant properties, for which we wish to establish whether the maximal probability of reaching certain "unsafe" states is less than some bound.

7.1. Introduction to probabilistic real-time reachability

Given a probabilistic timed automaton $G=(\mathscr{S},\mathscr{L},\bar{s},\mathscr{X},inv,prob,\langle\tau_s\rangle_{s\in\mathscr{S}})$, let R be a set of nodes called the *target set*, let $\sqsupseteq\in\{\geqslant,>\}$, and let $\lambda\in[0,1]$ be the *target probability*. Then the *probabilistic real-time reachability problem* for G can be defined as the triple $(R,\sqsupseteq,\lambda)$, with the intuition that the answer to the problem is "YES" if and only if G can reach a state in the target set R with probability $\sqsupseteq\lambda$, and "NO" otherwise.

To formalize probabilistic real-time reachability, we first must weaken the notion of divergent adversaries of probabilistic timed structures to take account of reachable states. Intuitively, an adversary of $\mathscr{M}_G$ is *R-divergent* if the probability measure over its paths which either reach R or exhibit realizable behaviour is 1. For a state $\langle s, v\rangle$, we define $\mathsf{discrete}\langle s, v\rangle = s$.

Definition 26. Let A be an adversary of $\mathscr{M}_G$. A path $\omega \in Path^A_{\mathrm{ful}}$ is *R-divergent* if it is either divergent or there exists $i \in \mathbb{N}$ such that $\mathsf{discrete}(\omega(i)) \in R$. An adversary A of $\mathscr{M}_G$ is R-divergent if and only if

$$Prob\{\omega \mid \omega \in Path^A_{\mathrm{ful}} \text{ and } \omega \text{ is } R\text{-divergent}\} = 1.$$

Let $\mathscr{A}_{Rdiv}$ be the set of all R-divergent adversaries of $\mathscr{M}_G$.

Then the answer to the real-time reachability problem $(R, \sqsupseteq, \lambda)$ is "YES" if and only if there exists an R-divergent adversary $A \in \mathscr{A}_{Rdiv}$ such that:

$$Prob\{\omega \mid \omega \in Path^A_{\mathrm{ful}}\langle \bar{s}, \mathbf{0}\rangle \,\&\, \exists i \in \mathbb{N}.\mathsf{discrete}(\omega(i)) \in R\} \sqsupseteq \lambda$$

and "NO" otherwise. That is, the answer is "YES" if and only if an R-divergent adversary can be found for which the probability measure over paths which reach a node in R is $\sqsupseteq \lambda$. In contrast to non-probabilistic real-time reachability problems [14, 27], which consider finite paths only, in accordance with standard notions of reachability in probabilistic systems [15] we consider the reachability of states in *infinite* paths.

Consider the fact that probabilistic real-time reachability properties require that it is *possible* to reach a certain set of states with probability λ or greater. It then follows that we are able to verify properties which specify the *inevitability* of reaching a certain set of states with a given probability λ or *less*. Clearly, such a property is true if it is not possible to reach the set of states with probability greater than λ. Note that inevitability properties with which we associate a lower bound on probability, such as the bounded response property of Section 3.1, "with probability 0.99 or greater, a data packet will always be delivered within 5 time units", cannot be expressed as a probabilistic real-time reachability property introduced in this section. However, verification of such a property is possible using PTCTL and the region graph.

Special case (*Reachability of symbolic states*). As in the nonprobabilistic context (see [27]), the reachability problem for *symbolic states* of a probabilistic timed automaton $G = (\mathscr{S}, \mathscr{L}, \bar{s}, \mathscr{X}, inv, prob, \langle \tau_s\rangle_{s\in\mathscr{S}})$ can be reduced to a problem of reachability of nodes, without loss of generality. By a symbolic state we mean a pair of the form $\langle s, \zeta\rangle$, where $s \in \mathscr{S}$ and $\zeta \in \mathbf{Z}_{\mathscr{X}}$. The method is based on the addition of extra transitions to G which are enabled when the target symbolic state is reached, and lead to another node s', which we then regard as the target node. Formally, we can reduce the probabilistic real-time reachability problem for symbolic states of G, $(R_{\mathrm{symb}}, \sqsupseteq, \lambda)$, where $R_{\mathrm{symb}} = \{\langle s_1, \zeta_1\rangle, \ldots, \langle s_m, \zeta_m\rangle\}$ is a set of symbolic states, and $\sqsupseteq$ and λ have their usual meanings, to the standard real-time reachability problem on a *new* probabilistic

timed automaton G' in the following manner. Let $G' = (\mathscr{S} \cup \{s'\}, \mathscr{L}', \bar{s}, \mathscr{X}, inv', prob', \langle \tau'_s \rangle_{s \in \mathscr{S} \cup \{s'\}})$, where

- $\mathscr{L}'(s') = \emptyset$, $inv'(s') =$ `true`, and $prob'(s') = \{p\}$, where $p(s', \emptyset) = 1$ and $\tau'_{s'}(p) =$ `true`,
- for all $s \in \mathscr{S}$, let $\mathscr{L}'(s) = \mathscr{L}(s)$, $inv'(s) = inv(s)$, $prob'(s) = prob(s) \cup \{p_{\langle s,\zeta \rangle} \mid \langle s, \zeta \rangle \in R_{\text{symb}}\}$, where $p_{\langle s,\zeta \rangle}(s', \emptyset) = 1$, $\tau'_s(p_{\langle s,\zeta \rangle}) = \zeta$, and, for all other $p \in prob(s)$, $\tau'_s(p) = \tau_s(p)$.

The problem $(R_{\text{symb}}, \sqsupseteq, \lambda)$ on G is then reduced to the associated problem $(\{s'\}, \sqsupseteq, \lambda)$ on G'.

Application 1 (*Time bounded reachability*). In certain cases, we may be interested in the reachability of certain nodes either before or after a time deadline has expired. For example, recall the communication protocol of Section 3.1, and consider the property, "with probability 0.975 or greater, it is possible to correctly deliver a data packet within 5 time units". Problems of this type can be solved in our framework in the following manner. First, the probabilistic timed automaton of interest, G, is augmented with a single clock z, which is intended to count the total elapsed time of system execution. The clock z does not feature in any of the invariant or enabling conditions, or in any clock resets of the new probabilistic timed automaton, which will be denoted by G_{+z} (although observe that, as usual, z is 0 initially). Say that we are interested in the reachability of the set R of nodes in time $\sim c$, where $\sim \in \{<, \leqslant, =, \geqslant, >\}$ and $c \in \mathbb{N}$. Then this problem can be phrased as the reachability problem $(R^{\sim c}_{\text{symb}}, \sqsupseteq, \lambda)$ on symbolic states, where $R^{\sim c}_{\text{symb}} = \{\langle s, z \sim c \rangle \mid s \in R\}$. G_{+z} then undergoes a further transformation to G'_{+z} in the manner described in the previous paragraph, on which the standard probabilistic real-time reachability analysis is then performed.

Application 2 (*Invariance verification*). Another type of property of interest requires that the probabilistic timed automaton G does not leave a certain set of nodes $I \subseteq \mathscr{S}$ with a given probability or greater. For example, in Section 3.1, the property, "with probability 0.875 or greater, the system never aborts", falls into this category. Such *invariance verification* properties can be reduced to standard probabilistic real-time reachability problems in the following way. Let the probabilistic invariance verification problem $\mathscr{I}$ be the tuple $(I, \sqsupseteq, \lambda)$, where $\sqsupseteq$ and λ have their usual meanings. Then $\mathscr{I}$ reduces to the probabilistic real-time reachability problem $(\mathscr{S} \backslash I, \bar{\sqsupseteq}, 1 - \lambda)$, where $\bar{\sqsupseteq} = >$ if $\sqsupseteq = \geqslant$ and $\bar{\sqsupseteq} = \geqslant$ if $\sqsupseteq = >$. That is, we solve the problem of reachability of nodes that are not in the required invariant I with probability $1 - \lambda$ or greater. If the answer to the problem $(\mathscr{S} \backslash I, \bar{\sqsupseteq}, 1 - \lambda)$ is "YES", then the answer to $\mathscr{I}$ is "NO", otherwise the answer to $\mathscr{I}$ is "YES".

Reachability properties and PTCTL: In the nonprobabilistic context, property specification languages such as TCTL do not capture reachable states [27], for the simple reason that the former are evaluated over infinite, divergent paths, whereas the latter are defined by finite paths. This characteristic also transfers to the probabilistic context, albeit in a slightly different manner; that is, PTCTL is interpreted over divergent ad-

versaries, whereas reachability properties are interpreted over R-divergent adversaries. However, if *all* adversaries of the probabilistic timed structure $\mathscr{M}_G$ of G are divergent (recall that the property of divergence is stronger than that of R-divergence), then we can identify correspondences between PTCTL formulae and reachability properties. Without loss of generality, for a given set S of nodes, we extend the labelling function $\mathscr{L}$ of the probabilistic timed automaton G to $\mathscr{L}_S : \mathscr{S} \to 2^{\mathrm{AP} \cup \{a_S\}}$ where, for each $s \in \mathscr{S}$, $\mathscr{L}_S(s) = \mathscr{L}(s) \cup \{a_S\}$ if $s \in S$, and $\mathscr{L}_S(s) = \mathscr{L}(s)$ otherwise.

7.2. Probabilistic real-time reachability algorithm

7.2.1. Preliminaries

Before the probabilistic real-time reachability algorithm is introduced formally, we present the following concepts for representing and manipulating state sets of a probabilistic timed automaton. The concepts of *c-equivalence* and *c-closure* are defined in a similar manner to [27] (note that they are related to the extrapolation abstraction of [14]). Given $c \in \mathbb{N}$, two valuations v, v' are described as *c-equivalent* if

- for any clock $x \in \mathscr{X}$, either $v(x) = v'(x)$, or $v(x) > c$ and $v'(x) > c$; and,
- for any two clocks $x_1, x_2 \in \mathscr{X}$, $v(x_1) - v(x_2) = v'(x_1) - v'(x_2)$, or $v(x_1) - v(x_2) > c$ and $v'(x_1) - v'(x_2) > c$.

We define the *c-closure* of ζ, denoted by $\mathsf{close}(\zeta, c)$, to be the greatest zone $\zeta' \supseteq \zeta$ satisfying, for all $v' \in \zeta'$, there exists $v \in \zeta$ such that v and v' are c-equivalent. Intuitively, the c-closure of a zone is obtained by removing all of its boundaries which correspond to constraints which refer to integers greater than c. Observe that, for a given c, there are only a finite number of c-closed zones.

Consider a particular probabilistic timed automaton G. Recall a pair of the form $\langle s, \zeta \rangle$, where $s \in \mathscr{S}$ and $\zeta \in \mathbf{Z}_{\mathscr{X}}$ is a zone, is called a *symbolic state*. We say that a state $\langle s, v \rangle$ is *contained within* a symbolic state $\langle s', \zeta \rangle$, written $\langle s, v \rangle \in \langle s, \zeta \rangle$, if $s = s'$ and $v \in \zeta$.

The function close is extended to symbolic states, by defining $\mathsf{close}(\langle s, \zeta \rangle, c) = \langle s, \mathsf{close}(\zeta, c) \rangle$. We also overload the function discrete to the case of symbolic states; that is, for a symbolic state $\langle s, \zeta \rangle$, we define $\mathsf{discrete}\langle s, \zeta \rangle = s$.

For the purposes of the forward reachability algorithm, it is convenient to regard the discrete transitions of a probabilistic timed automaton G as defining *edges* between the nodes of G. Formally, an edge e is a tuple of the form $(s, s', X, p) \in \mathscr{S}^2 \times 2^{\mathscr{X}} \times \mu(\mathscr{S} \times 2^{\mathscr{X}})$. We define the *set E of edges* of the probabilistic timed automaton G such that $(s, s', X, p) \in E$ if and only if $p \in prob(s)$ and $p(s', X) > 0$. For any $s \in \mathscr{S}$, the set $\mathsf{out}(s)$ contains all edges of the form $(s, \_, \_, \_)$.

Our aim of forward exploration through the state space of a probabilistic timed automaton requires operations to return the successor states of all of the states in a particular set (where the set is represented as a symbolic state). More precisely, we introduce a *discrete successor* operation which, given an edge $e \in E$ and a symbolic state $\langle s, \zeta \rangle$, returns all of the states obtained by traversing e from a state in $\langle s, \zeta \rangle$. Similarly, our *time successor* operation on the symbolic state $\langle s, \zeta \rangle$ returns the set of

states which can be obtained from a state in $\langle s, \zeta \rangle$ by letting some time elapse. These two operations can be composed to define a generalized successor operation, called the *post* operation. For a given symbolic state $\langle s, \zeta \rangle$ and an edge $e \in E$, the post operation returns the set of states that can be obtained from $\langle s, \zeta \rangle$ by traversing the edge e and then letting time elapse. Note that we also parameterize the post operation by an integer $c \in \mathbb{N}$, and only compute the c-closure of symbolic states obtained by the time successor operation; in Section 7.2.2, this fact will be used to ensure the termination of our forward reachability algorithm.

Definition 27 (*Successor operations*). The *forward projection* of a zone $\zeta \in \mathbf{Z}_{\mathscr{X}}$ is defined to be the zone $\nearrow \zeta$ in $\mathbf{Z}_{\mathscr{X}}$, such that $v \in \nearrow \zeta$ if and only if $\exists t \in \mathbb{R}. v - t \in \zeta$.
For a symbolic state $\langle s, \zeta \rangle$:

Time successor: The *time successor* of $\langle s, \zeta \rangle$ is defined as $\mathsf{time\_succ}\langle s, \zeta \rangle \stackrel{\text{def}}{=} \langle s, \nearrow \zeta \cap inv(s) \rangle$.

Discrete successor: The *discrete successor* of $\langle s, \zeta \rangle$ with respect to the edge $e = (s, s', X, p)$ is defined as

$$\mathsf{disc\_succ}(e, \langle s, \zeta \rangle) \stackrel{\text{def}}{=} \langle s', ((\zeta \cap \tau_s(p))[X := 0]) \cap inv(s') \rangle.$$

Post: For a constant $c \in \mathbb{N}$ and an edge $e \in \mathsf{out}(s)$, the *post operation* is defined according to:

$$\mathsf{post}[e, c]\langle s, \zeta \rangle \stackrel{\text{def}}{=} \mathsf{close}(\mathsf{time\_succ}(\mathsf{disc\_succ}(e, \langle s, \zeta \rangle)), c).$$

Observe that all of the operations in Definition 27 preserve the convexity of zones [27].

7.2.2. *The algorithm* **ForwardReachability**

An algorithm for generating a finite representation of the state space of a probabilistic timed automaton for a given set of target states R is presented in Fig. 6. As in the case of similar algorithms in the nonprobabilistic context [13, 24], the algorithm searches forward through a reachable portion of the state space of the system by successively iterating the transition relation a finite number of times. Note that the introduction of probabilistic information in the discrete transition relation of probabilistic timed automata means that we now regard the portion of the state space that is computed by this method to be reachable *with nonzero probability*. Given that this set has been generated for the probabilistic timed automaton G, we can then obtain an upper bound on the maximal probability of computations of G reaching the target set; this is achieved by solving a linear programming problem on the generated state space, in the manner of [15, 16]. Therefore, our strategy consists of *two* distinct computation steps: firstly, generation of the state space which is reachable with nonzero probability from the initial state, and secondly, performing another computation on this state space to find the maximum probability relevant to our problem.

```
ForwardReachability(⟨s̄, ζ_0⟩, R){
c := c_max(G)
Z := ∅
Reached := ∅
Fringe := {close(time_succ⟨s̄, ζ_0⟩, c)}
repeat
  choose ⟨s, ζ⟩ ∈ Fringe
  Fringe := Fringe\{⟨s, ζ⟩}
  if s ∈ R then Reached := Reached ∪ {⟨s, ζ⟩}
  else
    for each e ∈ out(s) do
      let ⟨s', ζ'⟩ := post[e, c]⟨s, ζ⟩
      if ζ' ≠ ∅ and ⟨s', ζ'⟩ ∉ Z then
        Fringe := Fringe ∪ {⟨s', ζ'⟩}
      end if
    end for each
  end if
  Z := Z ∪ {⟨s, ζ⟩}
until Fringe = ∅
return Z, Reached
}
```

Fig. 6. The algorithm **ForwardReachability**.

One important difference between the algorithm in Fig. 6 and analogous algorithms in the nonprobabilistic, real-time literature, is the fact that the *on-the-fly* property of the latter algorithms is compromized in our context. This property refers to the fact that, if a symbolic state which reaches the target set is computed, then the algorithm can terminate immediately with a "YES" answer to the reachability problem. Such a strategy is insufficient for probabilistic timed automata. For example, consider the case in which we have found a path of symbolic states reaching the target set R, and which corresponds to the probability λ. Then it may be possible to find another path to R, thus increasing the probability of reaching this target set. Given the expense of the probability computation step, we opt to perform it only after all of the relevant symbolic states which are reachable with positive probability have been computed.

The portion of the state space that is generated by **ForwardReachability** takes the form of a set Z of symbolic states $\langle s, \zeta \rangle$. The set *Fringe* is used to represent the set of symbolic states that are reachable from the initial state with non-zero probability, but whose successors may not have been explored.

Given the probabilistic timed automaton $G = (\mathscr{S}, \mathscr{L}, \bar{s}, \mathscr{X}, inv, prob, \langle \tau_s \rangle_{s \in \mathscr{S}})$, and the target set $R \subseteq \mathscr{S}$, the sets Z and *Reached*, obtained by the algorithm **ForwardReach-**

ability, are now used to define a Markov decision process $\mathcal{N}_G^R = (Q_G^R, Steps_G^R)$ called the *zone graph*, which contains the information relevant to the probabilistic real-time reachability problem. As in Sections 4.1 and 6.2, we associate a notion of type with the transitions of the zone graph, and overload the notation type for this purpose.

Definition 28 (*Zone graph*). The zone graph $\mathcal{N}_G^R$ of the probabilistic timed automaton G with respect to R is the Markov decision process $(Q_G^R, Steps_G^R)$ such that $Q_G^R = Z$ is the set of symbolic states computed by **ForwardReachability**, and $Steps_G^R : Q_G^R \to \mathcal{P}_{fin}(\mu(Q_G^R))$ is a set of transitions of the following form:

Discretetransitions: For all $\langle s, \zeta\rangle \in Z \backslash Reached$ and $p \in prob(s)$ where $\zeta \cap \tau_s(p) \neq \emptyset$, there exists $\hat{p} \in Steps_G^R\langle s, \zeta\rangle$, such that, for each $\langle s', \zeta'\rangle \in Z$:

$$\hat{p}\langle s', \zeta'\rangle = \sum_{\substack{X \subseteq \mathcal{X} \,\&\\ \langle s', \zeta'\rangle = \mathsf{post}[(s,s',X,p),c]\langle s,\zeta\rangle}} p(s', X).$$

Let $\mathsf{type}(\hat{p}) = p$.

Self-loops: For every $\langle s, \zeta\rangle \in Z$, let $\hat{p}_{\text{loop}} \in Steps_G^R\langle s, \zeta\rangle$, where $\hat{p}_{\text{loop}}\langle s', \zeta'\rangle = 1$ if and only if $\langle s', \zeta'\rangle = \langle s, \zeta\rangle$. Let $\mathsf{type}(\hat{p}_{\text{loop}}) = \perp$.

The initial state of the zone graph is $\mathsf{close}(\mathsf{time\_succ}\langle \bar{s}, \zeta_{\mathbf{0}}\rangle, c)$, which we denote by $\bar{q}$. As in the standard manner for Markov decision processes [8, 6], we can define notions of paths, the function $Prob'_{\text{fin}}$ on finite paths, and the probability measure $Prob'$ on infinite paths. Furthermore, the notion of adversaries of Markov decision processes is also standard; we denote the set of all adversaries of the zone graph by $\mathscr{B}$.

Recall that, for a given $c \in \mathbb{N}$, there are a finite number of c-closed zones. Lemma 29 then follows from the $c_{\max}(G)$-closure of all symbolic states of $\mathcal{N}_G^R$, and the fact that the number of nodes is finite [14, 27].

Lemma 29. *For any probabilistic timed automaton G and target set $R \subseteq \mathscr{S}$, Q_G^R is finite.*

Observe that Lemma 29 also implies the termination of the algorithm. Note that, as any symbolic state is a union of regions, and that the algorithm only generates distinct symbolic states, the size of the zone graph, and therefore the complexity of the verification problem, is the same as that of the associated region graph only in the *worst case*.

7.2.3. Computation of the maximal reachability probability

Given that the zone graph of the probabilistic timed automaton G with respect to the target set R has been constructed, the probabilistic real-time reachability problem $(R, \sqsupseteq, \lambda)$ can be solved in the following way. First, observe that if $Reached = \emptyset$, then the forward search through the reachable state space has found that R is not reachable with positive probability, and therefore we conclude that the answer to the problem is "No" (the problem for which $\sqsupseteq = \geqslant$ and $\lambda = 0$ is meaningless, and

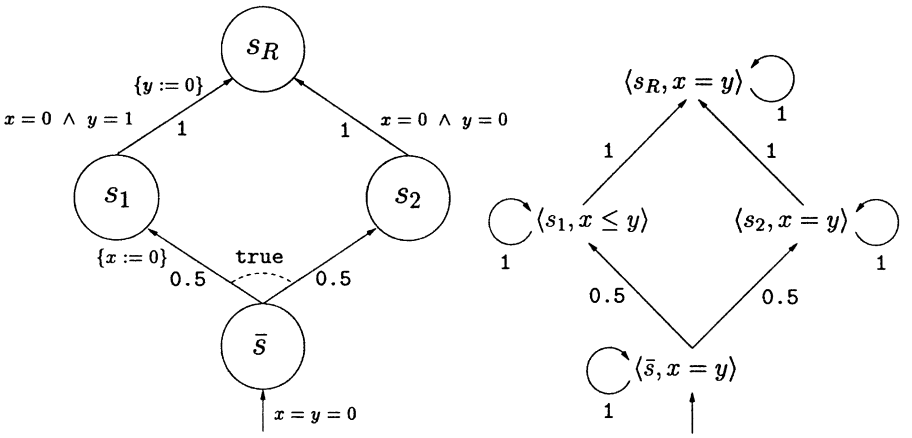

Fig. 7. The probabilistic timed automaton G_4 and the zone graph $\mathcal{N}_{G_4}^{\{s_R\}}$.

therefore is not considered here). If $Reached \neq \emptyset$, we propose to perform a maximal reachability probability computation on the state space of the zone graph, following established techniques for finite state Markov decision processes [11, 15, 16]. Unfortunately, the probability obtained via this approach may be greater than the probability of reaching the target set via *any* adversary of the probabilistic timed automaton.

This is illustrated by the example of the probabilistic timed automaton G_4, and its associated zone graph shown in Fig. 7. Say that the target set of interest consists of the single node s_R. Then it is clear that there are only two adversaries of G_4 (or, rather, of its probabilistic timed structure $\mathcal{M}_{G_4}$) such that there is a nonzero probability of reaching s_R. Intuitively, these correspond to two possible times at which the probability distribution associated with the initial state is chosen: either when $x = y = 0$, or when $x = y = 1$. In the former case, if the left-hand edge to s_1 is taken, then the outgoing edge of s_1 can never be taken, and so we must remain in this node; however, if the right-hand edge to s_2 is taken, then the outgoing edge to s_R can be selected immediately. The case for the selection of the transition of $\bar{s}$ when $x = y = 1$ is symmetrical. Therefore, for each of the two adversaries that have been described informally, the probability of reaching the target node s_R is 0.5. Now consider the zone graph $\mathcal{N}_{G_4}^{\{s_R\}}$ shown in Fig. 7 (the reader can verify that this is indeed the zone graph corresponding to the reachability problem in question). If, in $\langle \bar{s}, x = y \rangle$, the distribution to $\langle s_1, x \leqslant y \rangle$ and $\langle s_2, x = y \rangle$ is selected, and then, whichever symbolic state is chosen, a transition to $\langle s_R, x = y \rangle$ is made, then the probability of reaching a symbolic state for which the discrete part is s_R is 1. If the probabilistic real-time reachability problem in question is $(\{s_R\}, \geqslant, 0.7)$, then the maximal reachability probability that is computed on the zone graph will be greater than or equal to 0.7, whereas the answer to the probabilistic real-time reachability problem, as presented in Section 7.1, will be "No". Therefore, we advocate the strategy of returning an answer of "MAYBE", rather than "YES" if

the maximal reachability probability computed on the zone graph is $\sqsupseteq \lambda$, and "No" otherwise.

The maximal reachability probability problem on a Markov decision process is solved by reduction to an appropriate linear programming problem [11, 15, 16]. We now apply this solution to the Markov decision process $\mathcal{M}_G^R$, where the set of destination states (see [15]) is set equal to *Reached*. Let $\Pr(\mathsf{close}(\mathsf{time\_succ}\langle \bar{s}, \zeta_{\mathbf{0}}\rangle, c)$ be the maximal reachability probability value computed for the state $\mathsf{close}(\mathsf{time\_succ}\langle \bar{s}, \zeta_{\mathbf{0}}\rangle, c)$. Then, for a given probabilistic timed automaton G and a probabilistic real-time reachability problem $(R, \sqsupseteq, \lambda)$, the answer to the problem is

- "Maybe", if $\mathit{Reached} \neq \emptyset$ and $\Pr(\mathsf{close}(\mathsf{time\_succ}\langle \bar{s}, \zeta_{\mathbf{0}}\rangle, c)) \sqsupseteq \lambda$, and
- "No", otherwise.

Recall that invariance properties can be stated in terms of reachability properties, and that if the answer to the reachability property is "Yes" ("No") then the answer to the invariant property is "No" ("Yes", respectively). The introduction of the possibility of a "Maybe" result to the probabilistic real-time reachability problem has the following implications for the verification of invariance properties: if the answer to the reachability problem obtained from the invariance problem is "Maybe", then the answer to the invariance problem is also "Maybe"; however, if "No" is returned, then the answer to the invariance problem is "Yes". This leads us to conclude that our forward reachability method may be particularly appropriate for establishing the satisfaction of probabilistic, real-time invariance properties.

7.3. Example of probabilistic real-time reachability

Consider again the probabilistic timed automaton G_1 of Fig. 1. Say we are interested in the untimed, reachability property "with probability 0.9999 or greater, it is possible to correctly deliver a data packet" (we consider an untimed property for simplicity). This requirement can be specified in terms of the probabilistic real-time reachability problem $(\{ri\}, \geqslant, 0.9999)$, with the computation of the algorithm **ForwardReachability** taking the form given in Fig. 8. A typical iteration of the algorithm consists of the following steps: first, a symbolic state $\langle s, \zeta\rangle$ is taken from *Fringe*; second, for all edges that are enabled in $\langle s, \zeta\rangle$, the discrete successor is generated; third, the time successors of the discrete successors computed in the previous step are generated; finally, the sets *Fringe* and Z are adjusted, so that *Fringe* now contains the newly generated symbolic states (provided that they do not already exist in *Fringe* or Z), and $\langle s, \zeta\rangle$ is added to Z. Note that the second and third steps relate to the application of the $\mathsf{post}[.,.](.)$ operation, and that the $c_{\max}(G)$-closure of the time successors is implicit (as $c_{\max}(G) = 7$, we take the 7-closure of the time successors). The notation $s \to s'$, for nodes $s, s' \in \mathscr{S}$, refers to the edge from node s to s'. A special case is iteration 4, in which the node of the symbolic state $\langle ri, x = y\rangle$ is found to be in the target set R, and therefore $\langle ri, x = y\rangle$ is added to *Reached*.

The resulting zone graph $\mathcal{M}_{G_1}^{\{ri\}}$ is shown in Fig. 9 (we omit the self-loops that are associated with every symbolic state for simplicity). Observe that the most important

Initially: $Fringe = \{\langle hi, x = y \leqslant 3\rangle\}$, $Z = \emptyset$		
Iteration 1: take $\langle hi, x = y \leqslant 3\rangle$ from *Fringe*		
disc_succ	$hi \longrightarrow wr : \langle wr, \zeta_{\mathbf{0}}\rangle$	$hi \longrightarrow wi : \langle wi, x = 0 \wedge 2 \leqslant y \leqslant 3\rangle$
time_succ	$hi \longrightarrow wr : \langle wr, x = y \leqslant 1\rangle$	$hi \longrightarrow wi : \langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle$
$Fringe = \{\langle wr, x = y \leqslant 1\rangle, \langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle\}$, $Z = \{\langle hi, x = y \leqslant 3\rangle\}$		
Iteration 2: take $\langle wr, x = y \leqslant 1\rangle$ from *Fringe*		
disc_succ	$wr \longrightarrow ri : \langle ri, x = y \leqslant 1\rangle$	$wr \longrightarrow wi : \langle wi, x = y \leqslant 1\rangle$
time_succ	$wr \longrightarrow ri : \langle ri, x = y\rangle$	$wr \longrightarrow wi : \langle wi, x = y \leqslant 3\rangle$
$Fringe = \{\langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle, \langle ri, x = y\rangle, \langle wi, x = y \leqslant 3\rangle\}$, $Z = \{\langle hi, x = y \leqslant 3\rangle, \langle wr, x = y \leqslant 1\rangle\}$		
Iteration 3: take $\langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle$ from *Fringe*		
disc_succ	$wi \longrightarrow wr : \langle wr, \zeta_{\mathbf{0}}\rangle$	$wi \longrightarrow wi : \langle wi, x = 0 \wedge 4 \leqslant y \leqslant 6\rangle$
time_succ	$wi \longrightarrow wr : \langle wr, x = y \leqslant 1\rangle$	$wi \longrightarrow wi : \langle wi, x \leqslant 3 \wedge x + 4 \leqslant y \leqslant 7\rangle$
$Fringe = \{\langle ri, x = y\rangle, \langle wi, x = y \leqslant 3\rangle, \langle wi, x \leqslant 3 \wedge x + 4 \leqslant y \leqslant 7\rangle\}$, $Z = \{\langle hi, x = y \leqslant 3\rangle, \langle wr, x = y \leqslant 1\rangle, \langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle\}$		
Iteration 4: take $\langle ri, x = y\rangle$ from *Fringe*		
Add $\langle ri, x = y\rangle$ to *Reached*		
$Fringe = \{\langle wi, x = y \leqslant 3\rangle, \langle wi, x \leqslant 3 \wedge x + 4 \leqslant y \leqslant 7\rangle\}$, $Z = \{\langle hi, x = y \leqslant 3\rangle, \langle wr, x = y \leqslant 1\rangle, \langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle, \langle ri, x = y\rangle\}$		
Iteration 5: take $\langle wi, x = y \leqslant 3\rangle$ from *Fringe*		
disc_succ	$wi \longrightarrow wr : \langle wr, \zeta_{\mathbf{0}}\rangle$	$wi \longrightarrow wi : \langle wi, x = 0 \wedge 2 \leqslant y \leqslant 3\rangle$
time_succ	$wi \longrightarrow wr : \langle wr, x = y \leqslant 1\rangle$	$wi \longrightarrow wi : \langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle$
$Fringe = \{\langle wi, x \leqslant 3 \wedge x + 4 \leqslant y \leqslant 7\rangle\}$, $Z = \{\langle hi, x = y \leqslant 3\rangle, \langle wr, x = y \leqslant 1\rangle, \langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle, \langle ri, x = y\rangle, \langle wi, x = y \leqslant 3\rangle\}$		
Iteration 6: take $\langle wi, x \leqslant 3 \wedge x + 4 \leqslant y \leqslant 7\rangle$ from *Fringe*		
disc_succ	$wi \longrightarrow wr : \langle wr, \zeta_{\mathbf{0}}\rangle$	$wi \longrightarrow wi : \langle wi, x = 0 \wedge 6 \leqslant y \leqslant 7\rangle$	$wi \longrightarrow aa : \langle aa, \zeta_{\mathbf{0}}\rangle$
time_succ	$wi \longrightarrow wr : \langle wr, x = y \leqslant 1\rangle$	$wi \longrightarrow wi : \langle wi, x \leqslant 1 \wedge x + 6 \leqslant y \leqslant 7\rangle$	$wi \longrightarrow aa : \langle aa, x = y\rangle$
$Fringe = \{\langle wi, x \leqslant 1 \wedge x + 6 \leqslant y \leqslant 7\rangle, \langle aa, x = y\rangle\}$, $Z = \{\langle hi, x = y \leqslant 3\rangle, \langle wr, x = y \leqslant 1\rangle, \langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle, \langle ri, x = y\rangle, \langle wi, x = y \leqslant 3\rangle, \langle wi, x \leqslant 3 \wedge x + 4 \leqslant y \leqslant 7\rangle\}$		
Iteration 7: take $\langle wi, x \leqslant 1 \wedge x + 6 \leqslant y \leqslant 7\rangle$ from *Fringe*		
disc_succ	$wi \longrightarrow aa : \langle aa, \zeta_{\mathbf{0}}\rangle$	
time_succ	$wi \longrightarrow aa : \langle aa, x = y\rangle$	
$Fringe = \{\langle aa, x = y\rangle\}$, $Z = \{\langle hi, x = y \leqslant 3\rangle, \langle wr, x = y \leqslant 1\rangle, \langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle, \langle ri, x = y\rangle, \langle wi, x = y \leqslant 3\rangle, \langle wi, x \leqslant 3 \wedge x + 4 \leqslant y \leqslant 7\rangle, \langle wi, x \leqslant 1 \wedge x + 6 \leqslant y \leqslant 7\rangle\}$		
Iteration 8: take $\langle aa, x = y\rangle$ from *Fringe*		
disc_succ	$aa \longrightarrow aa : \langle aa, \zeta_{\mathbf{0}}\rangle$	
time_succ	$aa \longrightarrow aa : \langle aa, x = y\rangle$	
$Fringe = \emptyset$, $Z = \{\langle hi, x = y \leqslant 3\rangle, \langle wr, x = y \leqslant 1\rangle, \langle wi, x \leqslant 3 \wedge x + 2 \leqslant y \leqslant x + 3\rangle, \langle ri, x = y\rangle, \langle wi, x = y \leqslant 3\rangle, \langle wi, x \leqslant 3 \wedge x + 4 \leqslant y \leqslant 7\rangle, \langle wi, x \leqslant 1 \wedge x + 6 \leqslant y \leqslant 7\rangle, \langle aa, x = y\rangle\}$		
$Fringe = \emptyset$, therefore terminate		

Fig. 8. Computations of **ForwardReachability** on G_1.

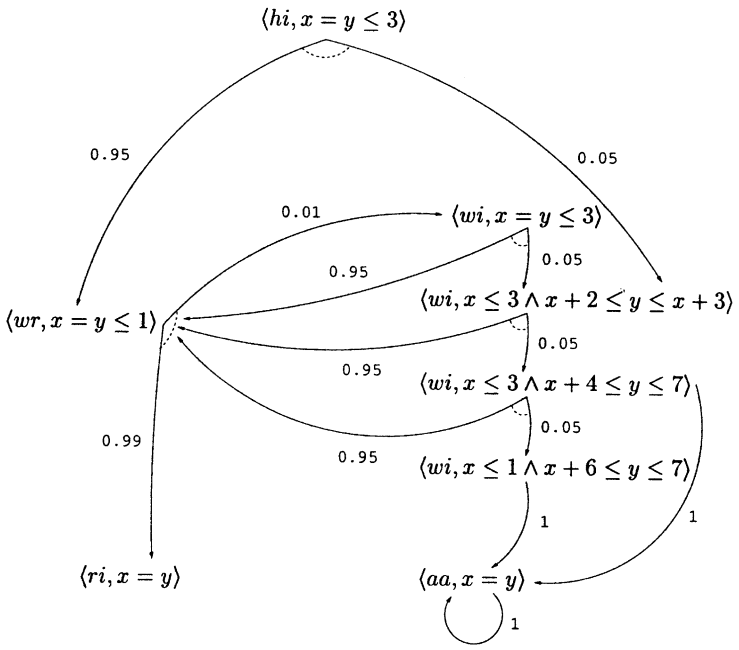

Fig. 9. The zone graph $\mathcal{N}_{G_1}^{\{ri\}}$.

nondeterministic choice made in this Markov decision process is featured in the symbolic state $\langle wi, x \leqslant 3 \wedge x + 4 \leqslant y \leqslant 7\rangle$, and that the single destination state of $\mathcal{N}_{G_1}^{\{ri\}}$ is the symbolic state in *Reached*, $\langle ri, x = y\rangle$. After application of a linear programming problem, we find that the maximal reachability probability value of reaching $\langle ri, x = y\rangle$ from the initial state $\langle hi, x = y \leqslant 3\rangle$ is 0.9998737375. As this value is less than that as 0.9999, we conclude that the answer to the reachability problem, "with probability 0.9999 or greater, it is possible to correctly deliver a data packet", is "No". Observe that the region graph of G_1 and the PTCTL property $[\texttt{true}\ \exists\mathcal{U}\ ri]_{\geqslant 0.9999}$, which is equivalent to the reachability property $(\{ri\}, \geqslant, 0.9999)$, consists of 101 reachable states, compared to the 8 states constructed by the forward reachability method of Fig. 6.

Note that it is straightforward to verify G_1 against a time bounded reachability property such as "with probability 0.975 or greater, it is possible to correctly deliver a data packet within 5 time units". As explained in Section 7.1, all that is required is to augment G_1 with an additional clock z, an extra node s', and an additional distribution in the node ri, which is enabled only when $z \leqslant 5$ and leads to the new target node s' with probability 1. We then perform reachability analysis on the new probabilistic timed automaton in a similar way to that described above. It also follows that verification of invariance properties of G_1, such as "with probability 0.875 or greater, the system never aborts" is possible.

7.4. Proof of the partial correctness of the probabilistic real-time reachability algorithm

We have shown that our forward reachability procedure will compute a maximal reachability probability on the zone graph that may be greater than the actual maximal probability on the probabilistic timed automaton G. However, we now show that the computed probability will not be less than the actual probability by proving that, for every adversary of the probabilistic timed structure $\mathscr{M}_G$ of G, there exists a sufficiently "similar" adversary of $\mathscr{N}_G^R$, where the notion of "similarity" means that the probability of reaching the target set R is the same for both adversaries. The proof proceeds in two steps: first, properties of single transitions of the zone graph $\mathscr{N}_G^R$ and the probabilistic timed structure $\mathscr{M}_G$ of G are studied, providing the basis of the second step, which concerns the paths and adversaries of $\mathscr{M}_G$ and $\mathscr{N}_G^R$.

7.4.1. Properties of single-step transitions of the zone graph

We commence by exploring the way in which a single transition of the probabilistic timed structure $\mathscr{M}_G$ of G may be mimicked by a transition of the zone graph $\mathscr{N}_G^R$, such that both transitions have the same type (as given by the functions denoted by type). Our concerns are twofold: first we show that, for any state $\langle s,v\rangle$ of $\mathscr{M}_G$ contained within a symbolic state $\langle s,\zeta\rangle$ of $\mathscr{N}_G^R$, any transition from $\langle s,v\rangle$ can be mimicked by a transition from $\langle s,\zeta\rangle$ of the same type; then we show that, for any target state $\langle s',v'\rangle$ of the transition from $\langle s,v\rangle$, the set of clock resets which can be used to obtain $\langle s',v'\rangle$ can be used to obtain a set of successor symbolic states of $\langle s,\zeta\rangle$, *each element of which contains* $\langle s',v'\rangle$.

Existence of transitions: First, we show that the existence of a transition from a state $\langle s,v\rangle$ of $\mathscr{M}_G$ implies the existence of a transition with the same type from any symbolic state of $\mathscr{N}_G^R$ which contains $\langle s,v\rangle$.

Lemma 30. *For any symbolic state* $\langle s,\zeta\rangle \in Q_G^R$, *state* $\langle s,v\rangle \in Q_G$ *such that* $\langle s,v\rangle \in \langle s,\zeta\rangle$, *and transition* $\langle s,v\rangle \xrightarrow{t,\tilde{p}} \langle s',v'\rangle$ *of* $\mathscr{M}_G$, *there exists a transition* $\langle s,\zeta\rangle \xrightarrow{\hat{p}} \langle s',\zeta'\rangle$ *of* $\mathscr{N}_G^R$, *such that* $\mathsf{type}(\tilde{p}) = \mathsf{type}(\hat{p})$ *and* $\langle s',v'\rangle \in \langle s',\zeta'\rangle$.

Proof. The correctness of this lemma follows from the definitions of post and of the zone graph $\mathscr{N}_G^R$ (Definitions 27 and 28). We have two cases, depending on whether $\mathsf{type}(\tilde{p}) = p$, for some $p \in prob(s)$, or whether $\mathsf{type}(\tilde{p}) = \perp$.

Case: $\mathsf{type}(\tilde{p}) = p$. Observe that there must exist an $X \subseteq \mathscr{X}$ such that both $v' = v[X := 0]$ and $p(s',X) > 0$. Furthermore, from $p(s',X) > 0$, there exists an edge $e = (s,s',X,p)$ in the set of edges of G. From the definition of $\mathsf{post}[.,.](.)$ (and the fact that the initial symbolic state of the zone graph is $\mathsf{close}(\mathsf{time\_succ}\langle \bar{s}, \mathbf{0}\rangle, c)$), all of the time successors of any state in $\langle s,\zeta\rangle$ (and all of the state that are c-equivalent to such time successors) are also in $\langle s,\zeta\rangle$. Because the transition $\langle s,v\rangle \xrightarrow{t,\tilde{p}} \langle s,v\rangle\, p$ and the fact that

$p = \mathsf{type}(\tilde{p})$ encode the information of letting t time units elapse from the state $\langle s, v\rangle$, and then choosing the distribution p, we conclude that $v + t \in \tau_s(p)$, and, since $v + t \in \zeta$, we have $\zeta \cap \tau_s(p) \neq \emptyset$. Furthermore, by the assumption of admissible targets of G, $((\zeta \cap \tau_s(p))[X := 0]) \cap inv(s') \neq \emptyset$. This gives us that $\mathsf{disc\_succ}(e, \langle s, \zeta\rangle) = \langle s', \zeta''\rangle$, where $\zeta'' \neq \emptyset$. As $v+t \in \tau_s(p)$, and $v' = (v+t)[X := 0]$, it must be the case that $v' \in \zeta''$, and therefore $\langle s', v'\rangle \in \langle s', \zeta''\rangle$. Next, because $\mathsf{close}(\mathsf{time\_succ}\langle s', \zeta''\rangle, c) \supseteq \langle s', \zeta''\rangle$, it must be the case that $\langle s', v'\rangle \in \mathsf{post}[e, c]\langle s, \zeta\rangle$. Then we let $\langle s', \zeta'\rangle = \mathsf{post}[e, c]\langle s, \zeta\rangle$ and $\langle s', v'\rangle \in \langle s', \zeta'\rangle$ as required.

Case: $\mathsf{type}(\tilde{p}) = \perp$. From Definition 9, it must be the case that $\langle s', v'\rangle = \langle s, v + t\rangle$. Furthermore, by Defintion 28, the only distribution available in $\langle s, \zeta\rangle$ with type $\perp$ is a self loop $\hat{p}_{\text{loop}}$; therefore, we mimic the transition $\langle s, v\rangle \xrightarrow{t, \tilde{p}} \langle s, v + t\rangle$ with the transition $\langle s, \zeta\rangle \xrightarrow{\hat{p}_{\text{loop}}} \langle s', \zeta'\rangle$, where $\langle s', \zeta'\rangle = \langle s, \zeta\rangle$. Because all of the time successors of any state in $\langle s, \zeta\rangle$, together with all of the states that are c-equivalent to any such time successor, are also in $\langle s, \zeta\rangle$, we have that $\langle s, v + t\rangle \in \langle s, \zeta\rangle$, which is equivalent to $\langle s', v'\rangle \in \langle s', \zeta'\rangle$. □

Targets of transitions: We now proceed to the main lemma of this section. Intuitively, we highlight the correspondence between transitions of the probabilistic timed structure $\mathscr{M}_G$ and the zone graph $\mathscr{N}_G^R$ by establishing the following facts. Consider the state $\langle s, v\rangle$, the symbolic state $\langle s, \zeta\rangle$ such that $\langle s, v\rangle \in \langle s, \zeta\rangle$, and the distributions $\tilde{p}, \hat{p}$ which are available in $\langle s, v\rangle$ and $\langle s, \zeta\rangle$, respectively, and for which $\mathsf{type}(\tilde{p}) = \mathsf{type}(\hat{p}) = p$. Then:

- every state of $\mathscr{M}_G$ which is in the support of $\tilde{p}$ is contained in a symbolic state in the support of $\hat{p}$; conversely, every symbolic state of $\mathscr{N}_G^R$ in the support of $\hat{p}$ contains a state in the support of $\tilde{p}$;
- furthermore, with the intuition that the probability $\tilde{p}\langle s', v'\rangle$ is equal to the sum of probabilities assigned by p to edges whose target is s' and which reset clocks in such a way as to result in the valuation v', then traversing each of these edges from the symbolic state will result in a target symbolic state $\langle s', \zeta'\rangle$ (not necessarily the same for each edge) which contains $\langle s', v'\rangle$.

First, consider the following facts concerning the effects of clock resets on valuations and zones. Given a valuation $v \in \mathbb{R}^{\mathscr{X}}$, a particular valuation $v' \in \mathbb{R}^{\mathscr{X}}$ may be obtained via more than one clock reset. For example, if $\mathscr{X} = \{x, y\}$, and $v(x) = v(y) = 0$, then $v[\{x\} := 0] = v[\{y\} := 0] = v[\{x, y\} := 0]$. However, note that the application of these reset sets to a given zone ζ may result in *different* zones. Consider the zone ζ given by the constraint $x \leqslant y$. Then $v \in \zeta$, yet $\zeta[\{x\} := 0]$ is defined by $0 = x \leqslant y$, whereas $\zeta[\{y\} := 0]$ is defined by $0 = y \leqslant x$, and $\zeta[\{x, y\} := 0]$ is defined by $x = y = 0$.

These concepts can be described formally in the following way.

Lemma 31. *Given* $\zeta \in \mathbf{Z}_{\mathscr{X}}$, $X_1, X_2 \subseteq \mathscr{X}$ *such that* $\zeta[X_1 := 0] = \zeta[X_2 := 0]$, *if* $v \in \zeta$, *then* $v[X_1 := 0] = v[X_2 := 0]$.

The lemma states that, if the application of two different resets to a zone ζ results in the same zone ζ', then the application of these resets to a valuation v contained in ζ must result in the same valuation v'. However, the converse of Lemma 31 (that, if $v[X_1:=0]=v[X_2:=0]$ and $v\in\zeta$, then $\zeta[X_1:=0]=\zeta[X_2:=0]$) is *not* necessarily true. Therefore, although multiple reset sets may result in the same valuation, these sets may result in different zones, although it is clear that the valuation will be contained within *all* such zones. Consider again the above example. Although $\zeta[\{x\}:=0]\neq\zeta[\{y\}:=0]\neq\zeta[\{x,y\}:=0]$, it is the case that $v'=v[\{x\}:=0]=v[\{y\}:=0]=v[\{x,y\}:=0]$ is such that $v'\in\zeta[\{x\}:=0]$, $v'\in\zeta[\{y\}:=0]$ and $v'\in\zeta[\{x,y\}:=0]$.

We now introduce three functions in order to reason about the relationship between valuations and zones.

- $\mathsf{SResets}:\mathbb{R}^{\mathscr{X}}\times\mathbb{R}^{\mathscr{X}}\to 2^{2^{\mathscr{X}}}$ is the function such that, for any $v,v'\in\mathbb{R}^{\mathscr{X}}$, $\mathsf{SResets}(v,v')\stackrel{\text{def}}{=}\{X\subseteq\mathscr{X}\mid v[X:=0]=v'\}$.
- Similarly, $\mathsf{ZResets}:\mathbf{Z}_{\mathscr{X}}\times\mathbf{Z}_{\mathscr{X}}\to 2^{2^{\mathscr{X}}}$ is the function such that, for any $\zeta,\zeta'\in\mathbf{Z}_{\mathscr{X}}$, $\mathsf{ZResets}(\zeta,\zeta')\stackrel{\text{def}}{=}\{X\subseteq\mathscr{X}\mid\zeta[X:=0]=\zeta'\}$.
- Finally, $\mathsf{NewZones}:\mathbf{Z}_{\mathscr{X}}\times 2^{2^{\mathscr{X}}}\to 2^{\mathbf{Z}_{\mathscr{X}}}$ is the function such that, for any $\zeta\in\mathbf{Z}_{\mathscr{X}}$ and $\mathscr{Y}\subseteq 2^{\mathscr{X}}$, $\mathsf{NewZones}(\zeta,\mathscr{Y})\stackrel{\text{def}}{=}\{\zeta'\in\mathbf{Z}_{\mathscr{X}}\mid\zeta'=\zeta[X:=0]$ for some $X\in\mathscr{Y}\}$.

The intuition underlying these functions is as follows. $\mathsf{SResets}(v,v')$ is the set of clock resets which, if applied to the valuation v, result in the valuation v', and $\mathsf{ZResets}(\zeta,\zeta')$ is the set of clock resets which, if applied to the zone ζ, result in the zone ζ' (note that $\mathsf{SResets}(v,v')$ and $\mathsf{ZResets}(\zeta,\zeta')$ may contain the empty clock reset $\emptyset$). $\mathsf{NewZones}(\zeta,\mathscr{Y})$ is the set of zones which can be obtained by the application of clock resets in $\mathscr{Y}$ to ζ.

The following lemma follows immediately from Lemma 31 and these definitions.

Lemma 32. *Let $\zeta\in\mathbf{Z}_{\mathscr{X}}$ be a zone.*

(1) *For any valuations $v,v'\in\mathbb{R}^{\mathscr{X}}$, and zone $\zeta'\in\mathbf{Z}_{\mathscr{X}}$, such that $v\in\zeta$ and $v'\in\zeta'$, then* $\mathsf{ZResets}(\zeta,\zeta')\subseteq\mathsf{SResets}(v,v')$.

(2) *For any distinct zones $\zeta',\zeta''\in\mathbf{Z}_{\mathscr{X}}$,* $\mathsf{ZResets}(\zeta,\zeta')\cap\mathsf{ZResets}(\zeta,\zeta'')=\emptyset$.

(3) *For any valuations $v,v'\in\mathbb{R}^{\mathscr{X}}$, and zone $\zeta'\in\mathbf{Z}_{\mathscr{X}}$, such that $v\in\zeta$ and $v'\in\zeta'$, then* $\bigcup\{\mathsf{ZResets}(\zeta,\zeta')\mid\zeta'\in\mathsf{NewZones}(\zeta,\mathsf{SResets}(v,v'))\}=\mathsf{SResets}(v,v')$.

Observe that part (2) of Lemma 32 implies that the elements of the set $\{\mathsf{ZResets}(\zeta,\zeta')\mid\zeta'\in\mathsf{NewZones}(\zeta,\mathsf{SResets}(v,v'))\}$ are themselves disjoint sets, and therefore the union in part (3) of the lemma is a *disjoint* union.

We now proceed to the main lemma of this section.

Lemma 33. *Let $\langle s,v\rangle\in Q_G$ be a state of $\mathscr{M}_G$, and $\langle s,\zeta\rangle\in Q_G^R$ be a symbolic state of $\mathscr{N}_G^R$ such that $\langle s,v\rangle\in\langle s,\zeta\rangle$. If $(t,\tilde{p})\in Steps_G\langle s,v\rangle$ and* $\mathsf{type}(\tilde{p})=p$ *for some $p\in\mu(\mathscr{S}\times 2^{\mathscr{X}})$, then for any $\langle s,v\rangle\in Q_G$ such that $\tilde{p}\langle s,v\rangle>0$:*

$$\tilde{p}\langle s',v'\rangle=\sum_{\zeta'\in\mathsf{NewZones}(\zeta\cap\tau_s(p),\mathsf{SResets}(v+t,v'))}\hat{p}(\mathsf{close}(\mathsf{time\_succ}\langle s',\zeta'\rangle,c)),$$

where $\hat{p}$ is the corresponding distribution of $Steps_G^R\langle s,\zeta\rangle$ given by Lemma 30.

Proof. From the definition of the distribution $\tilde{p}$ in Definition 9, and from the definition of $\mathsf{SResets}(.,.)$:

$$\tilde{p}\langle s', v'\rangle = \sum_{\substack{X\subseteq\mathscr{X}\,\&\\ (v+t)[X:=0]=v'}} p(s',X) = \sum_{X\in\mathsf{SResets}(v+t,v')} p(s',X). \tag{1}$$

Furthermore, by the definition of the distribution $\hat{p}$ in Definition 28, and from the definition of $\mathsf{ZResets}(.,.)$, for each $\zeta'\in\mathsf{NewZones}(\zeta\cap\tau_s(p),\mathsf{SResets}(v+t,v'))$:

$$\begin{aligned}\hat{p}(\mathsf{close}(\mathsf{time\_succ}\langle s',\zeta'\rangle,c)) &= \sum_{\substack{X\subseteq\mathscr{X}\,\&\\ \langle s,\zeta\rangle=\mathsf{disc\_succ}((s,s',X,p)\langle s,\zeta\rangle)}} p(s',X)\\ &= \sum_{X\in\mathsf{ZResets}(\zeta\cap\tau_s(p),\zeta')} p(s',X).\end{aligned}$$

Summing over all zones in $\mathsf{NewZones}(\zeta\cap\tau_s(p),\mathsf{SResets}(v+t,v'))$ gives the following equation:

$$\begin{aligned}&\sum_{\zeta'\in\mathsf{NewZones}(\zeta\cap\tau_s(p),\mathsf{SResets}(v+t,v'))} \hat{p}(\mathsf{close}(\mathsf{time\_succ}\langle s',\zeta'\rangle,c))\\ &\quad= \sum_{\zeta'\in\mathsf{NewZones}(\zeta\cap\tau_s(p),\mathsf{SResets}(v+t,v'))}\left(\sum_{X\in\mathsf{ZResets}(\zeta\cap\tau_s(p),\zeta')} p(s',X)\right).\end{aligned} \tag{2}$$

We now reconcile Eqs. (1) and (2). The following equation relies on the crucial observation of Lemma 32 that all of the elements of the set $\{\mathsf{ZResets}(\zeta\cap\tau_s(p),\zeta')\mid\zeta'\in\mathsf{NewZones}(\zeta\cap\tau_s(p),\mathsf{SResets}(v,v'))\}$ are themselves disjoint sets:

$$\begin{aligned}&\sum_{X\in\mathsf{SResets}(v+t,v')} p(s',X)\\ &\quad= \sum_{\zeta'\in\mathsf{NewZones}(\zeta\cap\tau_s(p),\mathsf{SResets}(v+t,v'))}\left(\sum_{X\in\mathsf{ZResets}(\zeta\cap\tau_s(p),\zeta')} p(s',X)\right).\end{aligned}$$

Therefore, from (1) and (2), after performing appropriate substitutions:

$$\tilde{p}\langle s', v'\rangle = \sum_{\zeta'\in\mathsf{NewZones}(\zeta\cap\tau_s(p),\mathsf{SResets}(v+t,v'))} \hat{p}(\mathsf{close}(\mathsf{time\_succ}\langle s',\zeta'\rangle,c)). \qquad \square$$

We now present an analogous result to Lemma 33 for the case in which $\mathsf{type}(\tilde{p})=\mathsf{type}(\hat{p})=\perp$, which follows immediately from the definition of the probabilistic timed structure and the zone graph (Definitions 9 and 28, respectively).

Lemma 34. *Let $\langle s,v\rangle\in Q_G$ be a state of $\mathscr{M}_G$, and $\langle s,\zeta\rangle\in Q_G^R$ be a symbolic state of $\mathscr{N}_G^R$ such that $\langle s,v\rangle\in\langle s,\zeta\rangle$. If $(t,\tilde{p})\in Steps_G\langle s,v\rangle$ such that $\mathsf{type}(\tilde{p})=\perp$ and $\hat{p}$*

is the corresponding distribution of $Steps_G^R\langle s,\zeta\rangle$ *given by Lemma* 30, *then* $\tilde{p}\langle s, v+t\rangle = \hat{p}\langle s,\zeta\rangle = 1$.

We conclude this section with the following observation. For a distribution $\tilde{p}$ of the probabilistic timed structure, there will be a finite number $l \in \mathbb{N}$ of states in the support of $\tilde{p}$; then, for a related distribution $\hat{p}$ of the zone graph, which is of the same type as $\tilde{p}$, there will be *at least* l symbolic states within the support of $\hat{p}$. Intuitively, the transitions of the zone graph can be though of as "probabilistically branching more than" the associated transitions of the probabilistic timed structure.

7.4.2. Properties of adversaries of the zone graph

The properties of Section 7.4.1 can be extended to reason about the correspondence between the paths and adversaries of both the probabilistic timed structure $\mathscr{M}_G$ and the zone graph $\mathscr{N}_G^R$.

Adversary construction process: The lemmas of Section 7.4.1 suggest the following result: for any adversary A of $\mathscr{M}_G$, an adversary B of $\mathscr{N}_G^R$ can be constructed. This construction process will proceed by starting from the initial states of $\mathscr{M}_G$ and $\mathscr{N}_G^R$, progressing along the length of paths of A and successively adding transitions to the partially constructed paths of B. In particular, for a path $\omega \in Path_{\text{fin}}^A\langle\bar{s},\mathbf{0}\rangle$, we construct a set of paths $\Pi_\omega \subseteq Path_{\text{fin}}(\bar{q})$ which mimic the transitions of ω.

We proceed by induction on the length of paths of $\omega \in Path_{\text{fin}}^A\langle\bar{s},\mathbf{0}\rangle$. Our aim is to construct the set of paths $\Pi_\omega \subseteq Path_{\text{fin}}(\bar{q})$, defining the choices of the adversary B as we proceed along their length. Our base case concerns paths of length 0; that is, paths comprising of only the initial states $\langle\bar{s},\mathbf{0}\rangle$ and $\bar{q}$ of $\mathscr{M}_G$ and $\mathscr{N}_G^R$, respectively. As $\bar{q} = \mathsf{close}(\mathsf{time\_succ}\langle\bar{s},\zeta_{\mathbf{0}}\rangle, c)$, it follows that $\langle\bar{s},\mathbf{0}\rangle \in \bar{q}$.

Next, consider any path $\omega \in Path_{\text{fin}}^A\langle\bar{s},\mathbf{0}\rangle$ of length $i+1$, then ω is of the form $\omega' \xrightarrow{t,\tilde{p}} \langle s, v\rangle$ for some path ω' of length i and $\langle s, v\rangle \in Q_G$. By induction we have constructed the set of paths $\pi \in \Pi_{\omega'}$, and we show how to extend this set to form Π_ω. We consider the following cases:

Case: *R has already been reached*: That is, there exists a $j \leqslant i$ such that $\mathsf{discrete}(\omega'(j)) \in R$. From the induction hypothesis (recall that this says that a state along a path ω' is contained within a symbolic state at the same point along the path $\pi' \in \Pi_{\omega'}$), we conclude that $\mathsf{discrete}(\pi'(j)) = \mathsf{discrete}(\omega'(j))$. From the definition of the zone graph (Definition 28), once a state of $\mathscr{N}_G^R$ has reached a node in R, then it loops in its current symbolic state. Therefore, regardless of the transition made by A after ω', we let $B(\pi') = \hat{p}_{\text{loop}}$ for all $\pi' \in \Pi_{\omega'}$, and define Π_ω as the set of paths $\{\pi' \xrightarrow{\hat{p}_{\text{loop}}} last(\pi') \mid \pi' \in \Pi_{\omega'}\}$.

Case: *R has not already been reached*: Consider any $\pi' \in \Pi_{\omega'}$ and say $\pi'(i) = \langle s', \zeta_{\pi'}\rangle$ and $\omega'(i) = \langle s', v'\rangle$. Then from our induction hypothesis and Lemma 30, there exists $\hat{p}_{\pi'} \in Steps_G^R\langle s', \zeta_{\pi'}\rangle$ such that $\mathsf{type}(\hat{p}_{\pi'}) = \mathsf{type}(\tilde{p})$ and we simply let $B(\pi) = \hat{p}_{\pi'}$.

Now, we must consider the preservation of the induction hypothesis by the transition; that is, we extend π' by a set of transitions for which we are certain that

the target symbolic state contains $\langle s, v\rangle$. Then, to construct the set Π_ω, we take the union of these extensions over all $\pi' \in \Pi_{\omega'}$.

If $\mathsf{type}(\tilde{p}) = \bot$, then by Lemma 34 we have $\langle s, v\rangle \in \langle s', \zeta_{\pi'}\rangle$ and we simply extend π' to a single path $\pi' \xrightarrow{\hat{p}_{\pi'}} \langle s', \zeta_{\pi'}\rangle$.

On the other hand, if $\mathsf{type}(\tilde{p}) = p$ for some $p \in \mu(\mathscr{S} \times 2^{\mathscr{X}})$, then by Lemma 30, there exists at least one symbolic state $\langle s, \zeta\rangle$ in the support of $\hat{p}_{\pi'}$ such that $\langle s, v\rangle \in \langle s, \zeta\rangle$. Furthermore, for all $\zeta \in \mathsf{NewZones}(\zeta' \cap \tau_{s'}(p), \mathsf{SResets}(v' + t, v))$, it must be the case that $v \in \zeta$, and therefore $\langle s, v\rangle \in \mathsf{close}(\mathsf{time\_succ}\langle s, \zeta\rangle, c)$. Then we extend the path π' to the set of paths of the form

$$\pi' \xrightarrow{\hat{p}_{\pi'}} \mathsf{close}(\mathsf{time\_succ}\langle s, \zeta\rangle, c)$$

such that $\zeta \in \mathsf{NewZones}(\zeta' \cap \tau_{s'}(p), \mathsf{SResets}(v' + t, v) \cap \{X \mid p(s, X) > 0\})$. Note that for any such ζ, by Definition 27 and our assumption of admissible targets: $\mathsf{close}(\mathsf{time\_succ}\langle s, \zeta\rangle, c) = \mathsf{post}[(s', s, X, p), c]\langle s', \zeta'\rangle$ for some $(s', s, X, p) \in \mathsf{out}(s')$.

Furthermore, repeating the construction procedure for all $\omega \in Path^A_{\mathrm{fin}}\langle \bar{s}, \mathbf{0}\rangle$ results in an adversary B on $\mathscr{N}^R_G$. We say that B is *constructed* from A. Next, we present the fundamental property of B: that the probability with which it reaches the target set R of nodes is equal to the probability of A reaching the set R.

Lemma 35. *Let A be an adversary of $\mathscr{M}_G$, and B be the adversary of $\mathscr{N}^R_G$ constructed from A. If $\omega \in Path^A_{\mathrm{fin}}\langle \bar{s}, \mathbf{0}\rangle$ and $\mathsf{discrete}(\omega(i)) \notin R$ for all $0 \leqslant i < |\omega|$, then*

$$Prob_{\mathrm{fin}}(\omega) = Prob'_{\mathrm{fin}}(\Pi_\omega).$$

Proof. The proof is by induction on the length of ω. If $|\omega| = 0$, then by definition of $Prob_{\mathrm{fin}}$ and $Prob'_{\mathrm{fin}}$, and construct of Π_ω: $Prob_{\mathrm{fin}}(\omega) = Prob'_{\mathrm{fin}}(\Pi_\omega) = 1$ as required.

Now suppose that the lemma holds for all paths of length at most k, and consider any path $\omega \in Path^A_{\mathrm{fin}}\langle \bar{s}, \mathbf{0}\rangle$ of length $k+1$ such that $\mathsf{discrete}(\omega(i)) \notin R$ for all $0 \leqslant i < |\omega|$. If $A(\omega^{(k)}) = (t, \tilde{p})$, then ω is of the form

$$\omega' \xrightarrow{t, \tilde{p}} \langle s, v\rangle$$

for some $\langle s, v\rangle \in Q_G$ such that ω' is of length k and $\mathsf{discrete}(\omega'(i)) \notin R$ for all $0 \leqslant i < |\omega'|$, and hence by induction we have

$$Prob_{\mathrm{fin}}(\omega') = Prob'_{\mathrm{fin}}(\Pi_{\omega'}). \tag{3}$$

For any $\pi' \in \Pi_{\omega'}$, let $last(\pi') = \langle s', \zeta_{\pi'}\rangle$ and let $\hat{p}_{\pi'} \in Steps^R_G\langle s', \zeta_{\pi'}\rangle$ be the distribution given by Lemma 30 and $\tilde{p}$. We now have the following two cases to consider.

Case: $\mathsf{type}(\tilde{p}) = \bot$. By construction $\Pi_\omega = \{\pi' \xrightarrow{\hat{p}_{\pi'}} \langle s', \zeta_{\pi'}\rangle \mid \pi' \in \Pi_{\omega'}\}$, and hence

$$\begin{aligned} Prob'_{\mathrm{fin}}(\Pi_\omega) &= \sum_{\pi' \in \Pi_{\omega'}} Prob'_{\mathrm{fin}}(\pi' \xrightarrow{\hat{p}_{\pi'}} \langle s', \zeta_{\pi'}\rangle) \\ &= \sum_{\pi' \in \Pi_{\omega'}} Prob'_{\mathrm{fin}}(\pi') \cdot \hat{p}_{\pi'}\langle s', \zeta_{\pi'}\rangle \quad \text{by definition of } Prob'_{\mathrm{fin}} \end{aligned}$$

$$
\begin{aligned}
&= \sum_{\pi' \in \Pi_{\omega'}} Prob'_{\mathrm{fin}}(\pi') \cdot \tilde{p}\langle s, v\rangle \quad \text{by Lemma 34} \\
&= \left(\sum_{\pi' \in \Pi_{\omega'}} Prob'_{\mathrm{fin}}(\pi') \right) \cdot \tilde{p}\langle s, v\rangle \quad \text{rearranging} \\
&= Prob_{\mathrm{fin}}(\omega') \cdot \tilde{p}\langle s, v\rangle \quad \text{by (3)} \\
&= Prob_{\mathrm{fin}}(\omega' \xrightarrow{t,\tilde{p}} \langle s, v\rangle) \quad \text{by definition of } Prob_{\mathrm{fin}} \\
&= Prob_{\mathrm{fin}}(\omega) \quad \text{by construction.}
\end{aligned}
$$

Case: $\mathsf{type}(\tilde{p}) = p$ for some $p \in \mu(\mathscr{S} \times 2^{\mathscr{Z}})$. Suppose $last(\omega') = \langle s', v'\rangle$, then by the definition of $Prob'_{\mathrm{fin}}$ and the construction of B we have

$$
\begin{aligned}
&Prob'_{\mathrm{fin}}(\Pi_\omega) \\
&\quad = \sum_{\pi' \in \Pi_{\omega'}} Prob'_{\mathrm{fin}}(\pi') \\
&\qquad \cdot \left(\sum_{\zeta \in \mathsf{NewZones}(\zeta_{\pi'} \cap \tau_{s'}(p), \mathsf{SResets}(v'+t, v))} \hat{p}_{\pi'}(\mathsf{close}(\mathsf{time\_succ}\langle s, \zeta\rangle, c)) \right).
\end{aligned}
$$

Furthermore, by Lemma 33 and the construction of B for any $\pi \in \Pi_\omega$:

$$
\tilde{p}\langle s, v\rangle = \sum_{\zeta \in \mathsf{NewZones}(\zeta_{\pi'} \cap \tau_{s'}(p), \mathsf{SResets}(v'+t, v))} \hat{p}_{\pi'}(\mathsf{close}(\mathsf{time\_succ}\langle s, \zeta\rangle, c)).
$$

Putting this together, we have

$$
\begin{aligned}
Prob'_{\mathrm{fin}}(\Pi_\omega) &= \sum_{\pi' \in \Pi_{\omega'}} Prob'_{\mathrm{fin}}(\pi') \cdot \tilde{p}\langle s, v\rangle \\
&= \left(\sum_{\pi' \in \Pi_{\omega'}} Prob'_{\mathrm{fin}}(\pi') \right) \cdot \tilde{p}\langle s, v\rangle \quad \text{rearranging} \\
&= Prob'_{\mathrm{fin}}(\Pi_{\omega'}) \cdot \tilde{p}\langle s, v\rangle \quad \text{by definition of } Prob'_{\mathrm{fin}} \\
&= Prob_{\mathrm{fin}}(\omega') \cdot \tilde{p}\langle s, v\rangle \quad \text{by (3)} \\
&= Prob_{\mathrm{fin}}(\omega' \xrightarrow{t,\tilde{p}} \langle s, v\rangle) \quad \text{by definition of } Prob_{\mathrm{fin}} \\
&= Prob_{\mathrm{fin}}(\omega) \quad \text{by construction}
\end{aligned}
$$

as required. □

Proposition 36. *Let A be an adversary of $\mathscr{M}_G$, and B be the adversary of $\mathscr{N}_G^R$ constructed from A. Then*

$$
\begin{aligned}
&Prob\{\omega \mid \omega \in Path^A_{\mathrm{full}}\langle \bar{s}, \mathbf{0}\rangle \,\&\, \exists i \in \mathbb{N}.\mathsf{discrete}(\omega(i)) \in R\} \\
&\quad = Prob'\{\pi \mid \pi \in Path^B_{\mathrm{full}}(\bar{q}) \,\&\, \exists i \in \mathbb{N}.\mathsf{discrete}(\pi(i)) \in R\}.
\end{aligned}
$$

Proof. The proof follows from Lemma 35 and the fact that the probability measures $Prob$ and $Prob'$ are induced from the functions $Prob_{\text{fin}}$ and $Prob'_{\text{fin}}$ respectively. □

Divergence on the zone graph: The following notion of divergence of paths of the zone graph agrees with the definition given for the nonprobabilistic case in [9]. We use the predicate $\mathsf{unbounded}(.,.)$ of [27] to express the unboundedness of a clock in a certain zone. Given a clock $x \in \mathscr{X}$ and a zone $\zeta \in \mathbf{Z}_{\mathscr{X}}$, we define

$$\mathsf{unbounded}(x,\zeta) \stackrel{\text{def}}{=} \begin{cases} \texttt{true} & \text{if } \forall t \in \mathbb{R}.\exists v \in \zeta.v(x) > t, \\ \texttt{false} & \text{otherwise.} \end{cases}$$

This predicate is now used to express the manner in which the value of clock x may become arbitrarily large before a certain discrete transition is executed. The intuition underlying the following definition of divergent paths on the zone graph is that, for each clock $x \in \mathscr{X}$, we require that x is either reset infinitely often, or, from some point on, x is unbounded. Because of the presence of reachable states, we define R-divergence in the usual way.

Definition 37 (*Divergent zone graph paths*). Let $\pi = \langle s_0,\zeta_0\rangle \xrightarrow{\hat{p}_0} \langle s_1,\zeta_1\rangle \xrightarrow{\hat{p}_1} \cdots$ be an infinite path of the zone graph.

Divergent: The path π is *divergent* if and only if, for each clock $x \in \mathscr{X}$, either:

(1) for every $i \in \mathbb{N}$, there exists $j \geqslant i$ and edge $e = (s_j, s_{j+1}, X, \mathsf{type}(\hat{p}_j)) \in \mathsf{out}(s_j)$ such that $x \in X$ and $\langle s_{j+1},\zeta_{j+1}\rangle = \mathsf{post}[e,c]\langle s_j,\zeta_j\rangle$, or

(2) there exists $i \in \mathbb{N}$ such that, for all $j \geqslant i$:
- if $\mathsf{type}(\hat{p}_j) = \perp$ then the predicate $\mathsf{unbounded}(x,\zeta_j)$ is true;
- if $\mathsf{type}(\hat{p}_j) \neq \perp$, then the predicates $\mathsf{unbounded}(x,\tau_{s_j}(\mathsf{type}(\hat{p}_j)))$ and $\mathsf{unbounded}(x,\zeta_j)$ are true.

R-divergent: The path π is *R-divergent* if and only if either:

(1) there exists $i \in \mathbb{N}$ such that $s_i \in R$, or

(2) π is divergent.

We now define divergent adversaries on the zone graph in the obvious manner.

Definition 38. An adversary B of the zone graph $\mathscr{N}_G^R$ is *R-divergent* if and only if

$$Prob'\{\pi \mid \pi \in Path^B_{\text{ful}} \text{ and } \pi \text{ is } R\text{-divergent}\} = 1.$$

Proposition 39. *For any R-divergent adversary A of $\mathscr{M}_G$, the adversary B of $\mathscr{N}_G^R$ that is constructed from A is R-divergent.*

Proof. We proceed by showing that, if the path ω of A is R-divergent, then all paths $\pi \in \Pi_\omega$ are also R-divergent (where Π_ω is the set of paths introduced in the construction procedure for B). Firstly, we consider the case in which there exists $i \in \mathbb{N}$ such that $\mathsf{discrete}(\omega(i)) \in R$; that is, the path ω reaches a target node. Then, by the construc-

tion procedure for B, for all paths $\pi \in \Pi_\omega$, $\mathsf{discrete}(\pi(i)) = \mathsf{discrete}(\omega(i))$. Therefore, $\mathsf{discrete}(\pi(i)) \in R$, and π is R-divergent.

Secondly, we consider the case in which ω does not reach the target set R. Consider the zone graph path $\pi \in \Pi_\omega$. Say that there exists a clock $x \in \mathscr{X}$ such that x is *not* reset from the ith transition in π onwards, for some $i \in \mathbb{N}$; that is, for the clock x, π does not satisfy part (1) of the definition of divergent zone graph paths. Then, as the path ω is divergent, and therefore corresponds to time passing above any bound, it must be the case that the value of x is not constrained by either an invariant condition or an enabling condition which is associated with a probability distribution of G taken at some point along ω from the ith transition onwards. Therefore, condition (2) of the definition of divergent zone graph paths must be satisfied by π, and π is then R-divergent. □

We conclude that, for the target set of nodes R, and for any R-divergent adversary A of the probabilistic timed structure of G, we can construct an R-divergent adversary of the zone graph $\mathscr{N}_G^R$ such that the probability of reaching a node in R are the same for both A and B.

Therefore, for the adversary of $\mathscr{M}_G$ with the maximal probability of reaching a node in R, there exists an adversary of $\mathscr{N}_G^R$ with the same probability of reaching R. Hence, the maximal reachability probability computed on $\mathscr{N}_G^R$ must be an upper bound on the maximal probability of reaching a node in R for $\mathscr{M}_G$.

8. Conclusions

We conclude with a brief analysis of the complexity of our methods. First, consider the PTCTL model checking method based on the region graph. The time complexity of PBTL model checking is polynomial in the size of the system (measured by the number of states and transitions) and linear in the size of the formula [6] (see also the recent improvement [4]). Since the translation from PTCTL to the extended PBTL has no effect on the size of the formula, it follows that the model checking for PTCTL against probabilistic timed systems will be polynomial in the size of the region graph and linear in the size of the PTCTL formula. Note that the addition of probability distributions to timed automata does not significantly increase the size of the region graph over the size of the nonprobabilistic region graph, and that the reset quantifier formulae we have added to PBTL can be handled in a straightforward manner.

Future work could address the potential inefficiencies of this method. Model checking of real-time systems is expensive, with its complexity being exponential in the number of clocks and the magnitude of their upper bounds (denoted by $c_{\max}(G)$ and $c_{\max}(\phi)$ in our presentation). However, this complexity relates to the region construction, which we established in Section 7 would only be constructed by the forward reachability algorithm in the worst case; in practice, the zone graph may be significantly smaller than the region graph. Note that forward reachability is not the only technique in the

field of timed automaton model checking which exhibits the characteristic of being typically more efficient than the region construction approach. For instance, the backwards reachability algorithm of [21] has this property, and has already been extended to the verification of probabilistic timed automata against properties of a fragment of PTCTL involving only existential quantification over adversaries [23]. Another approach concerns the computation of the coarsest bisimulation quotient of a probabilistic timed automaton. In our setting, this notion of bisimulation borrows concepts from both the *time-abstract* bisimulation of [29] and the *probabilistic* bisimulation of [25]. It is anticipated that judicious combination of the algorithms of [29, 5], which compute the coarsest bisimilarity quotient for timed automata and Markov decision processes, respectively, will provide the basis of a similar algorithm for probabilistic timed automata. It follows from the arguments of [27, 26] that such probabilistic time-abstract bisimilarity preserves PTCTL, implying that model checking for the full class of PTCTL properties will be possible on the bisimilarity quotient structure, which, as with structures obtained via forwards or backwards reachability, may be significantly smaller than the region graph.

As noted in [27, 28], the existence of behaviours of a timed automaton which are not time divergent corresponds to modelling errors, as no real-life system can block the progress of time. Therefore, [27, 28] present algorithms for the detection of such errors using forward reachability algorithms. Subsequent work could address the adaptation of these techniques to probabilistic timed automata, using our notion of probabilistic time divergence of Definition 8. Another potential avenue of research is to aid the modelling of complex probabilistic real-time systems by developing an adequate notion of parallel composition for probabilistic timed automata. This would permit the description of a system as a network of probabilistic timed automaton components executing in parallel.

Acknowledgements

We would like to thank Colin Stirling, the members of the British Council/DAAD ARC project 1031 “Stochastic Modelling and Verification” and the anonymous referees for many valuable comments and suggestions.

References

[1] R. Alur, C. Courcoubetis, D. Dill, Model-checking for probabilistic real-time systems, Automata, Languages and Programming: Proc. 18th ICALP, Lecture Notes in Computer Science, Vol. 510, 1991, pp. 115–126.
[2] R. Alur, C. Courcoubetis, D. Dill, Model-checking in dense real-time, Inform. and Comput. 104 (1) (1993) 2–34.
[3] R. Alur, D. Dill, A theory of timed automata, Theoret. Comput. Sci. 126 (1994) 183–235.
[4] C. Baier, On algorithmic verification methods for probabilistic systems, Habilitation Thesis, University of Mannheim, 1998.
[5] C. Baier, B. Engelen, M. Majster-Cederbaum, Deciding bisimilarity and similarity for probabilistic processes, J. Comput. System Sci. 60 (1) (2000) 187–231.

[6] C. Baier, M. Kwiatkowska, Model checking for a probabilistic branching time logic with fairness, Distributed Comput. 11 (1998) 125–155.
[7] J. Bengtsson, K. Larsen, F. Larsson, P. Pettersson, W. Yi, C. Weise, New generation of UPPAAL, Proc. Internat. Workshop on Software Tools for Technology Transfer, Aalborg, Denmark, July, 1998.
[8] A. Bianco, L. de Alfaro, Model checking of probabilistic and nondeterministic systems, in: Proc. Foundations of Software Technology and Theoretical Computer Science (FST& TCS), Lecture Notes in Computer Science, Vol. 1026, Springer, Berlin, 1995, pp. 499–513. LNCSD9, 0302-9743, Sat May 11 13:45:32 MDT 1996, ack-nhfb.
[9] A. Bouajjani, S. Tripakis, S. Yovine, On-the-fly symbolic model checking for real-time systems, 18th IEEE Real-Time Systems Symp. (RTSS'97), 1997.
[10] M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, S. Yovine, Kronos: a model-checking tool for real-time systems, in: A. Hu, M. Vardi (Eds.), Proc. 10th Internat. Conf. on Computer-Aided Verification (CAV'98), Lecture Notes in Computer Science, Vol. 1427, Springer, Berlin, 1998.
[11] C. Courcoubetis, M. Yannakakis, Markov decision processes and regular events, IEEE Trans. Automat. Control 43 (10) (1998) 1399–1418.
[12] P. D'Argenio, J.-P. Katoen, T. Ruys, J. Tretmans, The bounded retransmission protocol must be on time!, in: E. Brinksma (Ed.), Proc. of Tools and Algorithms for the Construction and Analysis of Systems, (TACAS'97), Lecture Notes in Computer Science, Vol. 1217, Springer, Berlin, 1997, pp. 416–431.
[13] C. Daws, A. Olivero, S. Tripakis, S. Yovine, The tool kronos, in: R. Alur, T.A. Henzinger, E.D. Sontag (Eds.), Hybrid Systems III, Lecture Notes in Computer Science, Vol. 1066, Springer, Berlin, 1996, pp. 208–219.
[14] C. Daws, S. Tripakis, Model-checking of real-time reachability properties using abstractions, in: B. Steffen (Ed.), Proc. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'98), Lecture Notes in Computer Science, Vol. 1384, Springer, Berlin, 1998.
[15] L. de Alfaro, Formal verification of probabilistic systems, Ph.D. Thesis Stanford University, Department of Computer Science, 1997.
[16] L. de Alfaro, Computing minimum and maximum reachability times in probabilistic systems, in: J. Baeten, S. Mauw (Eds.), Proc. Internat. Conf. on Concurrency Theory (CONCUR'99), Lecture Notes in Computer Science, Vol. 1664, Springer, Berlin, 1999, pp. 66–81.
[17] D. Dill, Timing assumptions and verification of finite-state concurrent systems, in: J. Sifakis (Ed.), Automatic Verification Methods for Finite State Systems, Lecture Notes in Computer Science, Vol. 407, Springer, Berlin, 1989.
[18] H. Gregersen, H.E. Jensen, Formal design of reliable real time systems, Master's Thesis, Department of Mathematics and Computer Science, Aalborg University, 1995.
[19] H. Hansson, B. Jonsson, A logic for reasoning about time and reliability, Formal Aspects Comput. 6 (5) (1994) 512–535.
[20] T. Henzinger, O. Kupferman, From quantity to quality, in: O. Maler (Ed.), Proc. Internat. Workshop on Hybrid and Real-time Systems (HART'97), Lecture Notes in Computer Science, Vol. 1201, Springer, Berlin, 1997, pp. 48–62.
[21] T.A. Henzinger, X. Nicollin, J. Sifakis, S. Yovine, Symbolic model checking for real-time systems, Inform. and Comput. 111 (2) (1994) 193–244.
[22] M. Kwiatkowska, G. Norman, R. Segala, J. Sproston, Automatic verification of real-time systems with discrete probability distributions, in: J.-P. Katoen (Ed.), Proc. 5th Internat. AMAST Workshop on Real-Time and Probabilistic Systems (ARTS'99), Lecture Notes in Computer Science, Vol. 1601, Springer, Berlin, 1999, pp. 75–95.
[23] M. Kwiatkowska, G. Norman, J. Sproston, Symbolic model checking of probabilistic timed automata using backwards reachability, Tech. Report CSR-00-1, University of Birmingham, 2000.
[24] K.G. Larsen, P. Pettersson, W. Yi, UPPAAL in a nutshell, Software Tools Technology Transfer 1 (1+2) (1997) 134–152.
[25] K.G. Larsen, A. Skou, Bisimulation through probabilistic testing, Inform. and Comput. 94 (1991) 1–28.
[26] R. Segala, N.A. Lynch, Probabilistic simulations for probabilistic processes, Nordic J. Comput. 2 (2) (1995) 250–273.
[27] S. Tripakis, L'Analyse Formelle des Systèmes Temporisés en Pratique, Ph.D. Thesis, Université Joseph Fourier, 1998.

[28] S. Tripakis, Verifying progress in timed systems, in: J.-P. Katoen (Ed.), Proc. 5th Internat. AMAST Workshop on Real-Time and Probabilistic Systems (ARTS'99), Lecture Notes in Computer Science, Vol. 1601, Springer, Berlin, 1999, pp. 299–314.

[29] S. Tripakis, S. Yovine, Analysis of timed systems based on time-abstracting bisimulations, in: R. Alur, T.A. Henzinger (Eds.), Proc. Eighth Internat. Conf. on Computer Aided Verification (CAV'96)", Lecture Notes in Computer Science, Vol. 1102, Springer, Berlin, 1996, pp. 232–243.